Dissertation

submitted

to the

Combined Faculty for the Natural Sciences and Mathematics

of

Heidelberg University, Germany

for the degree of

Doctor of Natural Sciences

Put forward by

Dipl.–Math. Martin Leonhard Felis

Born in:

Schwäbisch Hall in Baden–Württemberg

Oral examination:

July 22nd 2015

Modeling Emotional Aspects in Human Locomotion

Advisors:

Prof. Dr. Katja Mombaur
Prof. Dr. Alain Berthoz

Zusammenfassung

Die Forschung über emotionale Körpersprache wird seit über 200 Jahren von mehreren Disziplinen wie Psychologie, Neurologie und Biologie betrieben. Ein Großteil der Arbeiten basiert auf Untersuchungen der Kinematik der Bewegung, während die darunterliegende Dynamik weitgehend unbeachtet ist. In dieser Arbeit modellieren wir den menschlichen Gang als nichtlineares mehrphasiges Optimalsteuerungsproblem, um die Dynamik von emotionalen menschlichen Gehbewegungen zu untersuchen. Unser Ansatz basiert auf der effizienten Simulation der Dynamik von Mehrkörpermodellen, einem hochparametrisiertem mathematischen Model des menschlichen Gangapparates und einer direkten Mehrzielmethode, um die Dynamik von kinematischen markerbasierten Gangaufnahmen zu analysieren.

Modellierung des menschlichen Körpers mit Hilfe von dynamischen Mehrkörpersystemen führt zu komplizierten differentiell-algebraischen Gleichungen deren Herleitung computergestützter Methoden bedarf. Wir haben ein neues Softwarepaket erstellt, das die Modellierung und numerische Berechnung von kinematischen und dynamischen Größen von Mehrkörpersystemen in generalisierten Koordinaten, inklusive Modellierung von externen Kontakten und Unstetigkeiten bei Kontaktereignissen, ermöglicht. Die auftretenden Größen werden mit Hilfe von rekursiven Algorithmen berechnet, die auf Featherstone's 6-D räumlichen Algebra basiert. Unsere Softwarepaket ist speziell für die Verwendung in numerischen Optimalsteuerungsproblemen zugeschnitten und es wurde großes Augenmerk auf Ausnutzung von Strukturen und Wiederverwendung bereits berechneter Werte gelegt, um eine effiziente Auswertung der Größen zu ermöglichen. Dadurch erreichen wir eine Performanz, die optimierten computergenerierten Code erreicht und teilweise übertrifft.

Wir haben ein neues hochparametrisiertes 3-D Metamodell für den menschlichen Bewegungsapparat erstellt. Es ist ein dynamisches Starrkörpermodell, das auf kinematischen und dynamischen Parametern aus der Biomechanik basiert, und es erlaubt ein Anpassen der Segmentmaße, Positionen der Gelenke und Trägheitseigenschaften einzelner Segmente, so dass Modelle spezifisch für die aufgenommenen Probanden erstellt werden können. Für den Kontakt des Menschmodells mit dem Boden haben wir speziell für den menschlichen Gang ein nichtholonomes Kontaktmodell entwickelt, das die Ferse mit einer Kugel und den Vorderfuß mit einem Liniensegment beschreibt.

Bei der Abbildung von aufgenommenen Gehbewegungen auf Starrkörperbewegungen treten folgende Schwierigkeiten auf: unbekannte Gelenkzentren, redundante Markerbewegungen und sekundäre Bewegungsartefakte der Markerbewegungen durch Haut- und anderen Gewebeverschiebungen. In dieser Arbeit haben wir einen halbautomatischen Ansatz entwickelt, bei dem Gelenkzentren und virtuelle Modellmarker manuell angepasst und danach die Markerbewegungen mit Hilfe einer nichtlinearen Optimierung auf Starrkörperbewegungen abgebildet werden. Unser Ansatz ist unabhängig von dem verwendeten Motion-Capture-Markerset und bildet direkt auf den Koordinatenraum der Freiheitsgrade des Starrkörpermodells ab.

Wir haben zwei Arten vom mehrphasigen Optimalsteuerungsproblemen formuliert, um den menschlichen Gang zu modellieren: ein inverses Rekonstruktionsproblem und ein Gangsyntheseproblem. Beide enthalten die Differentialgleichungen des menschlichen Starrkörpermodells als Nebenbedingungen und ermöglichen Untersuchungen unterschiedlicher Fragestellungen.

Das Rekonstruktionsproblem berechnet die unbekannten Gelenkkräfte von rein kinematischen Bewegungsaufnahmen, in unserem Fall die von emotionalen Gehbewegungen. Die so berechneten Gelenkkräfte in Ober- und Unterkörper weisen emotionsspezifische Muster auf, die auch in den aufgenommenen Muskelaktivitäten zu finden sind. Dies bestätigt unser Modell und unseren Ansatz, Optimalsteuerungsprobleme als Werkzeug zur Analyse menschlicher Körpersprache zu verwenden, wie es bisher noch nicht möglich war.

Unsere Gangsyntheseformulierung erlaubt das Generieren von Gehbewegungen, basierend auf rein mathematischen und physikalischen Prinzipien und kann in Computeranimation und in der prädiktiven Ganganalyse verwendet werden. Wir haben verschiedene Gehbewegungen erzeugt, indem wir Zielfunktion und Gangparameter variiert haben. Langfristiges Ziel dieser Formulierung ist insbesondere die Identifizierung der Optimalitätskriterien für emotionale Gehbewegungen. Dies soll durch Lösen eines hierarchisches Optimalsteuerungsproblems in zukünftigen Arbeiten untersucht werden.

Abstract

The study of emotional body language has been the effort of many scientists for more than 200 years, from areas such as psychology, neuroscience, biology, and others. A lot of work has focused on the analysis of the kinematics, while the study of the underlying dynamics is still largely unexplored. In this thesis we model human walking as a nonlinear multi-phase optimal control problem to investigate the dynamics of full-body emotional expressions in human locomotion. Our approach is based on rigid multibody dynamics, a highly parameterized mathematical model of the human locomotion system, and the direct multiple-shooting method to analyze the dynamics of recorded kinematic motion capture data.

Modeling the dynamics of a human rigid multibody model results in a set of highly complex differential algebraic equations that require automated methods to derive and evaluate. We created a new rigid multibody dynamics software package to model and numerically evaluate kinematic and dynamic quantities of rigid multibody systems expressed in generalized coordinates, including modeling of external contacts and discontinuities arising from contact events. Our package evaluates components of the equation of motion for multibody systems using recursive algorithms that are based on Featherstone's 6-D spatial algebra notation. Our package is specifically tailored for the use in numerical optimal control and carefully designed to exploit sparsities and reduction of redundant computations by selectively reusing computed values. By doing so we are able to achieve and partially exceed performance that is otherwise only available with source code generation modeling approaches.

We created a highly parameterized 3-D meta model for the human locomotion system. This rigid multibody model is based on biomechanical data for kinematic and inertial parameters and enables us to create subject-specific dynamic models by adjusting segment dimensions, joint locations, and inertial parameters. To describe the contact between the human model and the ground, we created a non-holonomic rigid body contact model specifically for human walking movements that approximates the foot geometry using a sphere for the heel and a line segment at the ball of the foot during forefoot contact.

Transforming motion capture marker data to rigid multibody motion is a difficult problem due to unknown joint centers, redundant marker movements, and non-rigid movement of markers as a result of skin and tissue movement. In this thesis, we developed and implemented a semi-automatic method in which we manually adjust the model to approximate the recorded subject and then compute joint angles by solving a non-linear least-squares optimization problem. Our approach is independent of the used motion capture marker set and directly maps onto the joint space of the model.

We formulate two types of multi-phase optimal control problems for human walking: an inverse reconstruction problem and a gait synthesis problem that both have the differential equations of the rigid multibody dynamics as a constraint and can be used for different purposes.

The reconstruction problem computes the unknown joint actuations from purely kinematic motion capture data. Applied to the recorded motion capture data, the reconstructed joint actuations show emotion specific features that are also found in the recorded muscle activity. This validates our model and approach to use optimal control problems as a tool to study emotional body language in a new way.

Our gait synthesis formulation allows the generation of walking motions solely based on mathematical and physical principles. It can be applied in computer animation, robotics, and predictive gait analysis. We have generated a wide range of motions by adjusting objective function and gait parameters. A long-term goal of this formulation is to investigate optimality criteria of emotional walking motions. For this, we aim to use hierarchical optimal control problems in our future works.

Acknowledgements

The research of this thesis was conducted at the research group *Optimization in Robotics and Biomechanics (ORB)* at the *Interdisciplinary Center for Scientific Computing (IWR)* and the Faculty of Mathematics and Computer Science at Heidelberg University.

Exploring bodily expressions of emotions using numerical optimal control methods and rigid multibody models has been a fascinating, exhausting, daring, thrilling, challenging, surprising, tedious, exciting, stimulating, frustrating, but most of all a wonderful experience for me. If it hadn't been for my supervisors *Prof. Dr. Katja Mombaur* from the Heidelberg University and *Prof. Dr. Alain Berthoz* from the Collège de France in Paris, I would not have had the opportunity to work on this topic.

I am deeply thankful to *Prof. Dr. Katja Mombaur* for her support, encouragement, and trust over the last years. Without her encouragement I would not have dared to undertake such an endeavor and I am very thankful for having had this possibility. Her guidance and experience in both optimization and biomechanical modeling have greatly contributed to the success of this thesis.

I am also indebted to *Prof. Dr. Alain Berthoz* and his endless reservoir of ideas of which unfortunately only a fraction I was able to put into practice in this thesis. I also thank him for his openness towards the mathematical concepts and methods developed in this thesis and the patience it takes until the results become visible.

I want to thank *Dr. Hideki Kadone* and *Nicolas Briand* who helped with the motion capture experiments. Furthermore I want to thank the anonymous participants of the motion capture recordings. I also want to thank *Stefan Breunig*, who helped with the creation of the online survey and the numerous participants of it.

I want to thank the *Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences (HGS)* funded by the *Deutsche Forschungs Gemeinschaft (DFG)* for both financial support and the scientifically stimulating environment. Furthermore I want to gratefully acknowledge the financial support from the European FP7 projects ECHORD (grant no. 231143), and KOROIBOT (grant no. 611909).

Being part of the *ORB* research group has been a great pleasure and I am thankful for my time with this wonderful composition of people, notably *Dr. Kathrin Hatz, Khai-Long Ho-Hoang, Alexander Schubert, Marina Horn, Henning Koch, Manuel Kudruss*, and *Benjamin Reh*, who witnessed almost all model iterations. I further want to thank *Dr. Debora Clever and Eike Fokken* for reading parts of the manuscript. I am also indebted to *Dr. Manish Sreenivasa* for giving highly valuable input on biomechanical modeling and analysis. Furthermore I want to thank *Sarah Otto* for her support in administrative tasks.

I want to thank the research group *Simulation and Optimization* of *Prof. Dr. Dr. h.c. mult. Hans Georg Bock* for giving me the possibility to work with their powerful software package MUSCOD-II, which I almost always enjoyed working with. Furthermore I want to thank *Dr. Johannes P. Schlöder* for his openness to share his wealth of knowledge of numerical simulation and optimization.

I am thankful to my parents *Ulrike* and *Ekkehard Felis* and siblings *Benjamin, Sebastian,* and *Clara* whose trust and support I can always count on. I further want to thank *Michel Toppin* who (rightfully) told me that everything is going to be okay.

I want to thank my wife Katrina whom I am thankful for every moment past, present, and future, and also our baby daughter Junia, whose explorations of emotional expressions are wonderful to watch.

To my brother Sebastian

He aroused my interest in advanced Mathematics by telling me that mathematicians are able to think in 12 dimensions.

I was 7, he was 10, and for $n = 12$ he was right.

Contents

Introduction

In our everyday life, we experience and are exposed to emotions. Be it through advertisements that suggest that we may buy *Happiness* or interactions with other humans. The question *"What is an emotion?"* is both simple to pose and difficult to answer. Are emotions distinct mental states? Bodily expressions? A result of our environment? A result of *our perception* of our environment? Are emotions necessary to live? Wouldn't it be better if we just used clear rational decisions to structure our lives?

Emotions are not simple states of mind that are distinct from rational thinking. Rather the contrary: emotions play an important role in resolving complex decisions for which no rational answer is available (Damasio, 1994; Berthoz, 2003).

A link between emotion and body was already suggested by scientists in the 19^{th} century (Darwin, 1872; James, 1884). Indeed, there is scientific evidence that the Facial Feedback Hypothesis, which states that an emotional facial expression influences how we perceive situations, is true (Strack et al., 1988). Various studies furthermore suggest that facial expressions of *Happiness, Anger, Disgust, Fear, Sadness,* and *Surprise* are universal (Ekman and Friesen, 1971). An important utility in the study of facial expressions is the Facial Action Coding System (FACS) (Hjortsjö, 1969; Ekman and Friesen, 1978), which is a standardized method to describe facial expressions based on movement of individual facial muscles.

However, emotions are also expressed using the body and facial expressions alone may not be sufficient to recognize emotions correctly. The study reported in Aviezer et al. (2012) showed that the bodily expressions governed the emotional rating of pictures of athletes at peak expressions in winning or loosing events. Neuroscientific experiments show that the perception of emotional body postures activate brain areas that are associated with action and movement de Gelder et al. (2004). This suggests that for understanding of emotions a holistic approach is required that considers not only the brain and the face, but the whole body.

Psychophysical experiments have consistently shown that emotional affect can be recognized from recorded walking motions, even for abstract visualizations such as dynamic point light displays (Atkinson et al., 2004). These indicate emotion specific patterns in these movements, which were explored, e.g. using machine learning (Omlor and Giese, 2007; Roether et al., 2009) and systematic analysis (Hicheur et al., 2013a) by only using kinematic motion capture data. The findings were also verified by modulating a previously neutral movement by modulating it, e.g. in terms of walk velocity and posture parameters such as head and trunk angles.

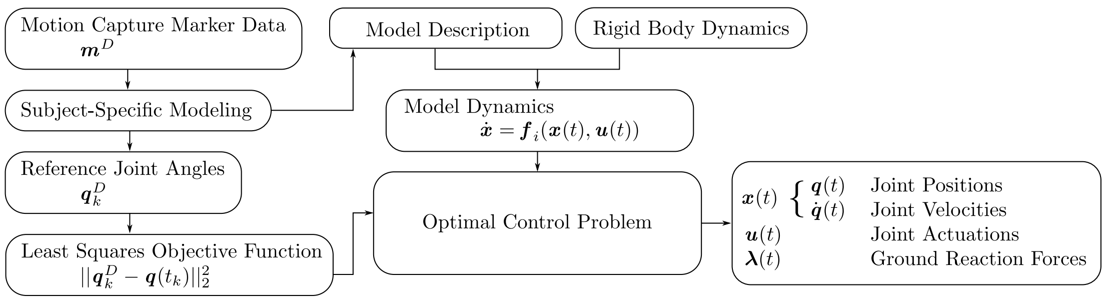
Most research efforts so far have been focused on kinematic data, such as marker movements or joint angles. A common method for the study of facial expressions is the study of muscle activity in the form of electromyographic data (EMG). For bodily expressions this is much less explored. Recently, new studies were performed on EMG data to find emotion specific muscle activity patterns in upper-body muscles (Huis in 't Veld et al., 2014) and also including back and leg muscles (Huis in 't Veld et al., 2014a). To date no study of the dynamics of emotional movement has been published.

Analysis of dynamics of human movement is an important method in biomechanics in general and specifically in gait analysis. There is a wide range of models that are used to analyse and describe human gait: from simple mechanical systems (Blickhan, 1989; Seyfarth et al., 2002; Kuo, 2007) to complex multibody models, which model hundreds of individual bodies and muscles (Delp et al., 1990; Nakamura et al., 2005; Hamner et al., 2010). The basic models already describe certain effects found in human walking and running (Geyer et al., 2006) while the musculoskeletal models can be used to reconstruct individual muscle contributions. To analyze a walking motion with a musculoskeletal models one has to provide not only the motion itself but also the ground reaction force, which requires additional measurement devices and imposes restrictions as to where the feet are placed during the experiment.

Scope of this Thesis

In this thesis we are modeling the dynamics of human gait as a nonlinear multi-phase optimal control problem of a underactuated hybrid differential-algebraic system. The optimal control approach we develop here allows us to reconstruct all dynamic quantities from kinematic motion capture recordings without the need of additional data such as ground reaction forces. We achieve this by modeling the human body as a dynamic 3-D rigid multibody system and minimize the deviation of the human model from recorded motion capture data while the equation of motion for rigid multibody systems is fulfilled for the human model. We solve the resulting highly non-linear multi-phase optimal control using the direct multiple-shooting method (Bock and Plitt, 1984). A schematic of our approach is shown in Figure 1.

An important aspect that we consider here is that we incorporate subject-specific differences in the model. A common approach to this is to use data from biomechanical literature (de Leva, 1996; Dumas et al., 2007) and create models based on the total weight and height of the subject. All other parameters such as segment lengths are then scaled uniformly based on the given weight and height. However, this approach leads to models that only coarseley represent the recorded subject as the segment lengths relative to body height is highly individual. To address this we formulate a highly parameterized rigid multibody model that can be adjusted specifically for each subject to closely match their kinematic properties.



Motion Capture Marker Data $\boldsymbol{m}^D$
Model Description
Rigid Body Dynamics
Subject-Specific Modeling
Model Dynamics $\dot{\boldsymbol{x}} = \boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t))$
Reference Joint Angles $\boldsymbol{q}_k^D$
Optimal Control Problem
Least Squares Objective Function $||\boldsymbol{q}_k^D - \boldsymbol{q}(t_k)||_2^2$
$\boldsymbol{x}(t)$ $\boldsymbol{q}(t)$ Joint Positions
$\dot{\boldsymbol{q}}(t)$ Joint Velocities
$\boldsymbol{u}(t)$ Joint Actuations
$\boldsymbol{\lambda}(t)$ Ground Reaction Forces

Figure 1: Schematic of our gait reconstruction approach.

We apply our dynamic reconstruction optimal control problem to a set of full-body motion capture sequences of human walking motions that express specific emotions. Additionally to the movement data the recordings also contain information of muscle activity in form of electromygraphic (EMG) data. The reconstructed joint torques of the different emotional expressions show emotion specific features that are also found in the EMG data, which validates our model as a new approach for analyzing human movement.

We further demonstrate how our human gait model can be used for full-body 3-D gait synthesis for which a schematic is shown in Figure 2. Based on the assumption that human movements are optimal in some sense (Alexander, 1984, 1997) we can generate walking motions for a wide range of optimization criteria that could be used for virtual characters or humanoid robots. Similar approaches have been used to compute optimal platform diving (Koschorreck and Mombaur, 2009), and human running (Mombaur, 2008; Schultz and Mombaur, 2010). As the dynamics of the full 3-D human model is considered, one can also apply our formulation to investigate how the movement changes when parameters of the model, the objective function, or the postural constraints such as movement range of individual limbs change. Also unknown model parameters (e.g. joint stiffness and damping) may be identified using our formulation. It furthermore allows testing hypotheses as to *why* movements are performed using our highly detailed human model, which is not possible with the existing simplistic biomechanical models.

Our gait synthesis formulation is a crucial component when trying to identify optimality

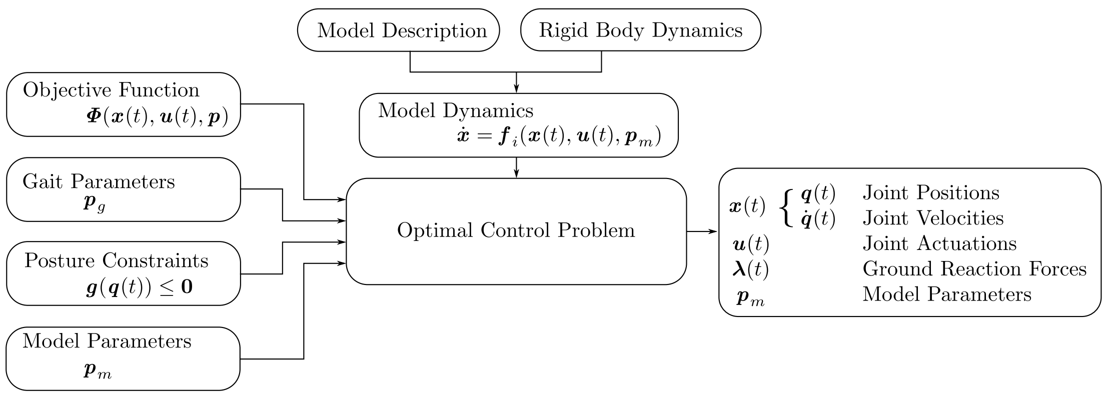

Model Description
Rigid Body Dynamics
Objective Function $\boldsymbol{\Phi}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p})$
Model Dynamics $\dot{\boldsymbol{x}} = \boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}_m)$
Gait Parameters $\boldsymbol{p}_g$
Optimal Control Problem
Posture Constraints $\boldsymbol{g}(\boldsymbol{q}(t)) \leq \boldsymbol{0}$
Model Parameters $\boldsymbol{p}_m$
$\boldsymbol{x}(t)$ $\boldsymbol{q}(t)$ Joint Positions
$\dot{\boldsymbol{q}}(t)$ Joint Velocities
$\boldsymbol{u}(t)$ Joint Actuations
$\boldsymbol{\lambda}(t)$ Ground Reaction Forces
$\boldsymbol{p}_m$ Model Parameters

Figure 2: Schematic of our full-body 3-D open-loop gait synthesis approach.

criteria for a given motion. Such *inverse* optimal control methods (or *hierarchic dynamic optimization* methods) have lately been used to *identify* the objective criteria of movement trajectories when walking from one target to another (Mombaur et al., 2010) or articulated movement of cerebral palsy patients (Hatz, 2014).

Contributions of this Thesis

A Highly Efficient Rigid-Body Dynamics Implementation During this thesis we created RBDL - the Rigid Body Dynamics Library. It is a self-contained model-based package for modeling and simulating dynamics of constrained rigid multibody systems specifically tailored for use in an optimal control context. It contains implementations of state-of-the-art algorithms of reduced-coordinate rigid multibody dynamics computations and further allows to model collisions and contact handling with the environment. It is based on Roy Featherstone's 6-D Spatial Algebra and allows us to efficiently evaluate quantities such as Jacobians, point accelerations, system angular momentum and others that are essential for the simulation and analysis of articulated motion. It uses recursive methods to perform these computations with an efficiency that is otherwise only achieved with source code generation tools. The library can be used in a variety of contexts such as simulation and real-time control of robotic systems and virtual characters.

A Method to Create Subject-Specific Models from Recorded Motion-Capture Data We discuss the transformation of spatial marker motions (i.e. motion capture marker data) onto rigid multibody models and the inherent problems. We solve the problem by a semi-automatic approach. The biomechanics literature provides tables to create dynamic models of humans for a given height and weight. Based on this we created HeiMan, a highly-parameterized model. For individual joints the relative locations of joint centers can be modified and are taken into account for inertial parameters of the segments. We created Puppeteer a motion capture visualization and rigid multibody modelling tool that allows us to visually adjust the model and then to transfer 3-D motion capture marker movements onto the joint movements of a rigid multibody model with high precision. The advantage of using subject-specific models is demonstrated.

Analysis of the Dynamics of Emotional Walking Motions using Numerical Optimal Control Methods We have formulated an optimal control problem that approximates motion capture data of human walking motions using dynamic subject specific models. The original motions are very closely reproduced using the model while being physically valid. We use this to compare the effects of different emotions on the level of joint torques of both lower and upper body of the model.

Open-Loop Synthesis of Full-Body 3-D Walking Movements We formulated and implemented a multi-phase optimal control problem that allows synthesis of open-loop full-body walking movements of a highly articulated model with 34 degrees of freedom. Without using motion capture data our approach allows generation of realistic movements that can for example be used to animate virtual characters and furthermore serve as a powerful method to study fundamentals of human walking.

A Validated Set of Emotional Motion Capture Data We have performed a motion capture experiment of five subjects that walked while expressing a set of emotions. We created an online survey in which the recorded data was transformed into dynamics point-light displays evaluated by more than one hundred participants. For each motion we have ratings concerning the expressiveness on a continuous scale. The results show that there is a high variance in the perceived expressiveness of the emotions. For two subjects however the emotions were very clearly recognized, which are thus suitable for the research of emotional body language.

Thesis Overview

Chapter 1 is dedicated to the fundamental descriptions of the biomechanics of the human gait and emotional body language. It introduces to the anatomical terms that are used throughout this thesis and gives an overview on existing research on emotional body language.

In the following Chapter 2 we describe the experiment that we conducted for this research. We performed simultaneous full-body motion capture and EMG recording of muscles on both upper and lower body. We also describe the online survey that we performed to verify the expressiveness and recognizability of the motions. The recorded EMG signals are presented and analyzed for emotion specific characteristics.

Chapter 3 describes the rigid multibody dynamics formulation and implementation that we created for this thesis to model the dynamics of the human body. The formulation is based on Featherstone's 6-D spatial algebra, which is a concise notation for rigid body movements and dynamics. The computations themselves are based on recursive algorithms that allow for a much greater ease of use compared to symbolic code generation. As the dynamics computations take up most of the computation time during the optimizations we have taken great care to efficiently implement the dynamics computations by exploiting sparsities in the 6-D operators and Jacobians. We discuss how we model contacts and collisions of the model with the ground and present a performance analysis of our implementation with symbolically generated code.

In the following Chapter 4 we describe how we transform the motion capture 3-D marker movement onto rigid multibody models and the mathematical model of the human body that we use in the optimization. We describe the inherent difficulties of transforming motion capture 3-D marker data onto articulated rigid body movements and how we solve this problem using a semi-automatic approach. We present HeiMan, a highly parameterized rigid multibody model we created based on biomechanical data that we adjust to obtain subject-specific models. We describe the robust nonlinear inverse kinematics method we use to compute joint angles from the motion capture data. We furthermore present the foot model that we use to describe the interaction of the model with the ground and also show how we embed the model dynamics as an ordinary differential equation.

In Chapter 5 we present the nonlinear multi-phase optimal control formulations that we developed to model the dynamics of human walking. We describe the direct multiple-shooting method that we use to solve this problem. We present two variations of the problem: the reconstruction problem to analyse a given motion and the synthesis problem that allows generation of open-loop human like 3-D walking motions. For both formulations we describe constraints and the individual phase descriptions. We also discuss how we construct start values for the optimizations.

Chapter 6 then presents the numerical results of the thesis. We show how our approach using subject specific modelling is advantageous both in terms of accuracy and required step count of the inverse kinematics when computing rigid multibody motions. We show the reconstructed joint actuations and compare it with the recorded EMG data. We show that emotion specific characteristics that are found in the EMG data are also found in the reconstructed joint actuations, which validates our approach. Further we present the numerical results for different walking gaits that we obtained using the gait synthesis formulation. We present and discuss gaits that we obtained for different objective functions and prescribed gait parameters. We present a similarity measure for human movement data and compare the synthesized gaits with the recorded emotional motion capture motions, by means of similarity, ground reaction forces, and elevation angles.

Part I

Preliminaries and Experiment

Chapter 1

Human Walking and Emotional Body Language

The study on emotional body language is an active research area that is being studied by many different disciplines such as biology, psychology, neuroscience, robotics, biomechanics, and others. As this thesis is concerned with emotional expressions during walking we describe in this chapter the fundamental concepts of biomechanical gait analysis and existing research on the study of emotional body language. We introduce anatomical terms and concepts that are commonly used in biomechanical movement descriptions and gait analysis. Individual phases of gait are described along with kinematic coordination patterns that are found in human locomotion. We present models that classify emotions in a set of basic emotions that are found across cultures and compare research of emotions using facial and bodily expressions. We conclude the chapter with recent results in the study of full-body emotional expressions.

1.1 Biomechanics of the Human Gait

1.1.1 Anatomic Planes and Terms

Anatomical terms and conventions make a clear description of general human movements possible. In Figure 1.1 we show the main anatomical planes and the names of the principal axes. When discussing the right or left side of the human body we follow the convention used in medical descriptions where the right (or left) side of a person is always the right (or left) side as seen from the person's point of view.

Anatomic Planes

The *Sagittal* plane splits the human body in left and right half and is parallel to the forward axis, which is also called the *Sagittal* axis. The *Coronal* or *Frontal* plane splits the human body in front (anterior) and back (posterior) part. The *Transverse* or *Axial* plane is parallel to the ground and the transversal axis.

Anatomic Axes

The *Sagittal* axis is the intersection of the Sagittal and Transverse plane, the *Transverse* axis the intersection of the Transverse and Coronal plane, and the *Longitudinal* axis the intersection of the Sagittal and Coronal plane.

One should note here that we use the term longitudinal axis also for the principal length of a segment, e.g. in Neutral pose, the longitudinal axis of the foot is parallel to the Sagittal axis of the human body, whereas the longitudinal axis of the elbow is parallel to the longitudinal axis of the human body. From the context it should be clear whether we mean the local axis of the segment or that of the human body.

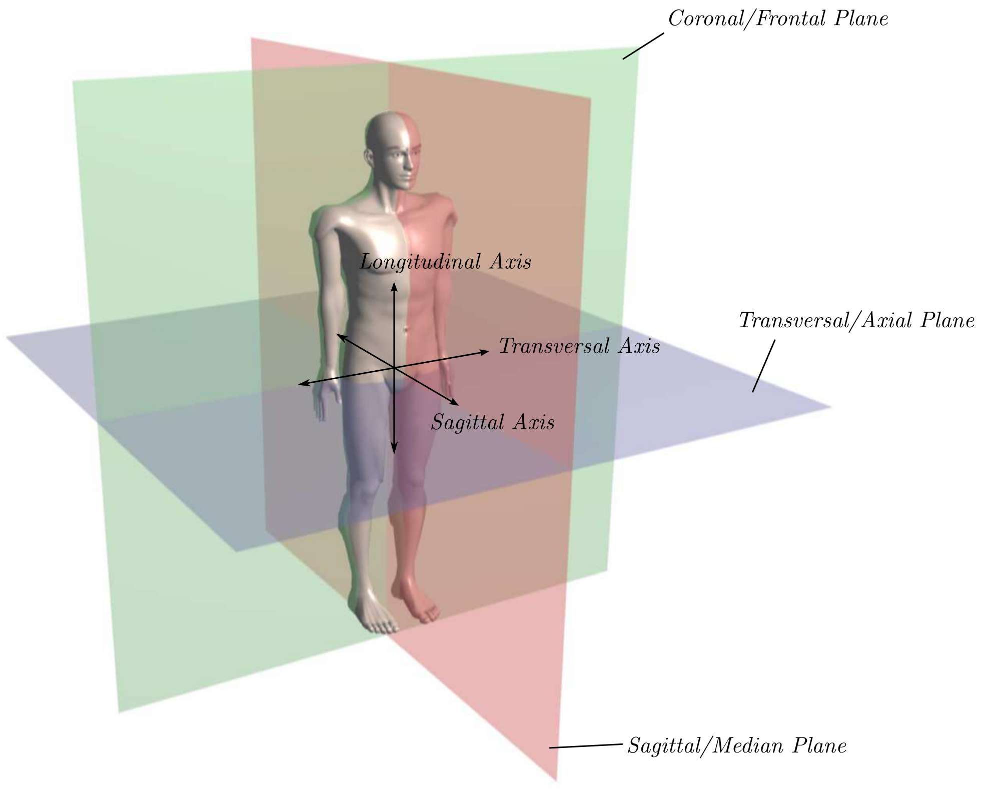


Figure 1.1: Anatomical planes and -axes of the human body. The shown posture is also called the *Neutral pose*.

Rotations

Rotations of limbs have specific names depending on the limb, the axis of rotation and rotation direction.

Extension/Flexion: Rotations around the Transversal axis are called *Flexions* if the rotation causes the angle to decrease, e.g. for a human that stands in the neutral pose when the thigh is lifted, or the knee is bent. *Extension* is the rotation opposite to a flexing motion, e.g. rotating the head backwards. Rotating the feet downwards is also called *Plantar Flexion* and the rotation upwards *Dorsi Flexion*.

Abduction/Adduction: The rotation around the Sagittal axis that causes a limb to move away from the Sagittal plane is called an *Abduction*, e.g. lifting an arm sideways. Rotating a joint such that the connected limb comes closer to the Sagittal plane is called an *Adduction*.

Internal/Medial Rotation: A rotation of a joint around the longitudinal axis towards the body center is called a *Internal* rotation, whereas the opposite rotation is called a *Medial* rotation.

Pronation/Supination: Rotation of the arms outwards, i.e. such that the thumbs point away from the body is called *Supination*. The same term is used when the feet are rotated inwards with toes pointing toward the other foot. The opposite movement is called *Pronation*.

Directions

Additionally to rotations there are anatomical terms for directions that we use in this thesis. These are:

Anterior/Posterior: An atomical point is said to be *Anterior* if it is located towards the front side of the coronal plane, e.g. the eyes are located at the *Anterior* of the head. The opposite direction of *Anterior* is *Posterior*.

Proximal/Distal: To refer to the ends of a limb one can use the terms proximal and distal. The *Proxmial* end of a limb is the one that is closer to the center of the body. E.g. the *Proximal* end of the shank is at the knee, whereas its *Distal* end is the ankle.

Medial/Lateral: Directions in the coronal plane are denoted by *Medial* and *Lateral*. *Medial* is used to refer to something that is towards the center along the transverse axis, whereas *Lateral* is used to refer to something that is at the side of the body away from the center.

1.1.2 Gait Analysis

Human walking is a form of locomotion in which the entire body is displaced forward by shifting the weight repeatedly from one foot to the other. Unlike running there is always at least one foot in contact with the ground. Another difference to running is that for a short duration during the gait cycle both feet are in contact with the ground.

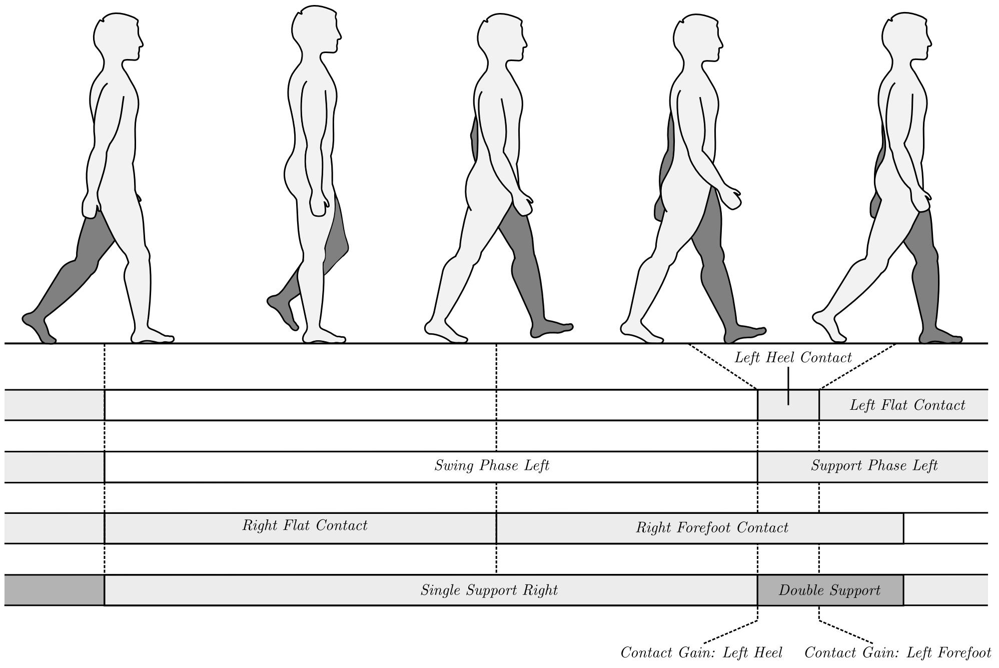


Figure 1.2: Overview of the gait phases and the constraint sets *RF* Right Flat, *RT* Right Toes, *RT LH* Right Toes Left Heel, *RT LF* Right Toes Left Flat. Ground collisions occur at *TD LH* (touchdown Left Heel) and *TD LT* (touchdown Left Toes).

Gait analysis is concerned with the description and analysis of walking motions. The analysis includes the determination of various gait parameters, such as as step length, step duration, walking velocity, contact forces, but also qualitative assessment of motions using recordings such as video recordings or motion capture methods. The latter allow recording of 3-D motion data e.g. by reconstructing the 3-D movement of markers that are attached to the human body using multiple cameras.

In the following we describe aspects of gait analysis that are relevant to this thesis. For a detailed description of the gait cycle we refer to Perry and Burnfield (2010).

Phases of Gait

A leg is said to be in the so-called *support phase* when the foot is in contact with the ground, otherwise it is in the *swing phase*. If both feet are in contact with the ground it is called *double-support phase* whereas if only one foot is in contact it is called *single-support phase*. A schematic is shown in Figure 1.2.

The human gait cycle can be described in different ways. One way is to look at the alternation of support- and swing-phase and to define the start of the cycle as the contact gain e.g. the contact gain of the right foot, which is followed by the right support-phase, the lift-off of the right foot and the following swing-phase. The cycle ends just before the next contact gain of the left foot. A single gait cycle would then be described as right support phase followed by the right swing phase. One should note that the gait cycles of the left and right leg do overlap.

An alternate view on the gait cycle is to consider the transitions between single- and double-support phases. I.e. the gait cycle starts with the right single-support phase right after the lift-off of the left foot. It transitions into the double-support phase with the contact gain of the left foot and continues until the right foot lifts off the ground. After that the cycle for the left foot begins with the same order of events.

Gait Parameters

Two distance parameters are often used in gait analysis: *step length* and *stride length*. The stride length is often defined as the distance of two consecutive heel strikes of the same foot, whereas the step length is the distance of two heel strikes of two distinct feet. In this thesis we often use an equivalent definition that uses the positions of the toes instead of the heels (see Figure 1.3). This latter view has the advantage that at moderate walking speeds time instances exist that allow a direct measurement of the step length. That both heels are in contact with the ground usually only occurs while standing or at very low walking speeds.

The step duration is the time interval between the start and the end of the step. Again, the duration between two consecutive heel strikes are commonly used to define this duration but it can also be defined as two consecutive lifts of the forefoot. Similarly, the stride duration is the time duration between two heel strikes of the same foot.

Walking velocity or walking speed is defined as the distance traveled, divided by duration, e.g. the step length divided by step duration or stride length divided by stride duration.

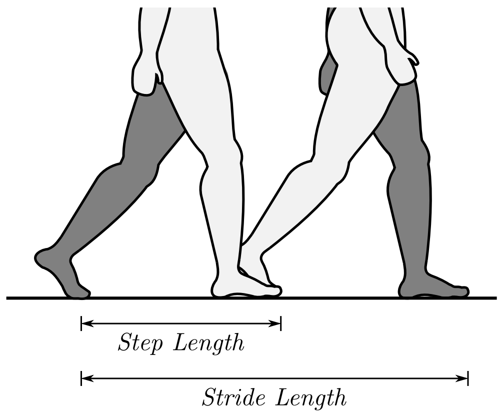


Figure 1.3: Gait parameters step length and stride length.

1.1.3 Coordination Patterns in Human Gait

Human walking is highly coordinated on multiple levels. First, human gait is cyclic, i.e. after a complete double-step the posture of the human is very close to that of the beginning. Furthermore, there is an anti-cyclic swinging of arms with respect to the movement of the legs. Experiments performed in robotics (Collins et al., 2009) suggest that the arm swinging reduces the ground reaction moment.

Another important form of coordination takes place in the balance of the head. Pozzo et al. (1990) investigated the linear and angular movement of the head in the sagittal plane during movements such as walking, walking in place, running in place and hopping. They show that during all tasks there is very little angular movement, e.g. during the walking tasks the angular movement was within 10°. Furthermore, the rotation of the head is anticyclic to the vertical translation of the head, i.e. if the head translates up, then the head flexes (rotates downward). This last phenomenon was found in all tasks. One interpretation of this is that these movements counteract disturbances in the vestibular system.

There is also a linear relationship of the angles of thigh, knee, and foot as shown in Borghese et al. (1996). The relationship however only seems to occur in the so-called *elevation angles*, i.e. the angles of the limbs in the global reference frame. The relative angles do not show the relationship. This phenomenon is sometimes also called the "law of intersegmental coordination", or "planar covariation", and occurs independently of the walking velocity. This was further investigated e.g. in Hicheur et al. (2006) and Ivanenko et al. (2008) for a different set of motions such as climbing stairs, air-stepping, walking with a bent back, or jumping. For all these motions the law of intersegmental coordination was found to hold.

1.2 Emotions and Emotional Body Language

Charles Darwin devoted extensive research efforts on emotions which he published 1872 in his book *The Expression of the Emotions in Man and Animals* Darwin (1872). He describes emotional expressions that can be found both in man and animal, which indicate that these are innate and that animals and humans have shared predecessors. A lot of research efforts have been undertaken over the last century. However, the answer to the question of William James "What is an Emotion" (James, 1884) still has no uncontested answer.

1.2.1 Classification of Emotions

More recently, attempts were made to classify emotions into a finite set of universal or basic emotions which are then combined for a wide range of emotions. Basic emotions are usually assumed to be innate and are triggered by events or objects and result in similar expressions or actions across cultures and species.

A well known example for a set of basic emotions was proposed by Ekman and Friesen (1978) who found that facial expressions of the emotions *Happiness*, *Sadness*, *Surprise*, *Fear*, *Disgust*, and *Anger* were recognized universally even for cultures that had not been exposed to television or other type of electrical media. Ekman and colleagues also found out that the facial expressions performed for these emotions used similar facial muscles across cultures (**?**).

Another model for basic emotions was proposed by Robert Plutchik using the *wheel of emotions* (Plutchik, 1980). It uses eight basic emotions, of which always two are paired in basic and basic opposite emotion: *Joy-Sadness*, *Trust-Disgust*, *Fear-Anger*, and *Surprise-Anticipation.* A wide range of emotions is then constructed by combining the basic emotions using different intensities.

One should note here that the concept of basic emotions is not universally accepted. A different approach is the so-called *appraisal approach*, which assumes that emotions are a result of an interpretation. I.e. an event or object is interpreted, which then causes an emotion. A bear that crosses my path in the wilderness is not automatically causing fear, instead the interpretation of the situation causes the emotion fear.

An overview on the history of emotion research and different views on emotions is for example given in Gendron and Feldman Barrett (2009).

Emotional Body Language

A large body of research on emotions is focused on the facial expressions of emotions. The *Facial Action Coding System (FACS)* (Ekman, 1992) e.g. provides a systematic standard to describe facial expressions using so-called *Action Units*. It allows describing a facial expression in a very detailed manner and can be used both in psychological experiments and in computer animation.

Other ways of expressing emotions are posture, gesticulation, positioning, and orientation relative to others, but also gaze. Walking velocity and the path chosen by a person that approaches another can vary in a myriad of ways and convey different types of information such as social status, previous and intended action, and many others. One can envision the slow and maybe fearful walk of a child that just destroyed a window with a soccer ball. Similarly, the excited running (and even jumping) of another that just won the most important basketball match of its life in the courtyard of a neighbour next door.

Research presented in Aviezer et al. (2012) showed that facial expressions alone are not sufficient to distinguish between strong positive or negative emotions. They used the expressions of world-class athletes during peak intensities such as winning and loosing and separated expression of body and face. Bodily expressions were successfully recognized but not facial expressions only. Furthermore, they superimposed facial and body images such that incongruent combinations such as "body expression win" and "facial expression loose" were created. For these images the body expressions seemed to govern the judgment of the participants.

One should note that the ability to read both facial and bodily expressions of emotion comes natural for most people, however there are various neural conditions such as autism spectrum disorders (ASD) which negatively affects this ability.

1.2.2 Perception of Bodily Expression of Emotion

Research on bodily expressions is generally performed differently than research on facial expressions. Studies on the latter often use static photos that include all the details of the face. Studies on the perception of bodily expressions often use videos of drastically reduced models such as point-light displays that only show points at certain anatomical landmarks e.g. points at head, shoulders, elbows, pelvis, knees, and ankles. One of the main reasons for using point-lights is the reduction of possible distractions, e.g. clothing, race, or gender, that would be included in full-light displays.

A study conducted by Gunnar Johansson in 1973 showed that walking motions shown as point-light displays were immediately recognized by all subjects of a study, even for short presentation times of only a second (Johansson, 1973). He also coined the term *Biological Motion* which refers to "biologically meaningful motion" such as the point-light display videos. The term is used to differentiate it from random motions of points.

Many characteristics of the recorded subject can be recognized from point-light displays of walking motions, such as gender (Mather and Murdoch, 1994). Also familiar persons can be identified (Cutting and Kozlowski, 1977). In Brownlow et al. (1997), emotions were successfully recognized in dynamic point-light displays of dancing movements.

Atkinson et al. (2004) compared the perception of emotions of full-light and point-light displays of either static stimuli (i.e. images) or dynamic stimuli (videos). Participants had to classify four sets of stimuli (full-light static, full-light dynamic, point-light static, point-light dynamic) of the emotions *Disgust*, *Fear*, *Anger*, *Happiness*, and *Sadness* that were performed by ten actors. Each set contained 150 stimuli and was classified using forced-choice by 36 participants. All emotions were recognized with a higher mean correct rate than chance. Overall the videos of full-light displays were recognized best (average score of 85.1, where 100 would be a perfect recognition), followed by videos of point-light displays (average score 76.2). The stills had an average score of 54.8 for full-light displays and 39.0 for point-light displays.

There also seem to be gender-specific differences in the perceptions of emotions. The study described in Krüger et al. (2013) used dynamic point-light displays of walking motions that expressed the emotions *Anger* and *Happiness*. The durations of these displays were $1.68s$ and participants had to classify the motions using a forced choice approach with the discrete options *Neutral*, *Happy*, and *Anger*. They show that males better recognized happy motions of female actors and that angry motions of male actors were better recognized by women.

1.2.3 Emotion Specific Motion Patterns

An extensive study on emotion specific motion patterns was conducted in Wallbott (1998). In this study 14 emotions (including emotions expressed in different intensities) were expressed by actors and recorded using full-light video cameras. The actors were free to gesticulate and no specific type of motion was described. The videos were then used to derive a set of codes such as "upper body collapsed", "head backward", "arms crossed in front of chest" and to analyze how significant these codes were for each emotion by manually annotating all videos. They suggest that expressions of emotions are not necessarily distinguished by a certain combination of codes but also, more importantly, by their intensities.

Omlor and Giese (2006) describe a blind source separation method that takes time shifts in the source signals into account, which is usually not considered in separation methods like principal component analysis (PCA) or independent component analysis (ICA). For given data $\boldsymbol{x}_i(t) \in \mathbb{R}^m$ they compute source signals $\boldsymbol{s}_j(t) \in \mathbb{R}^n$ with $n < m$, mixing weights $(\alpha_{i,j})_{i,j} \in \mathbb{R}^{n \times m}$ and time delays $(\tau_{i,j})_{i,j} \in \mathbb{R}^{n \times m}$ such that the original data is reconstructed by

$$\boldsymbol{x}_i(t) = \sum_{j=1}^{n} \alpha_{i,j} \boldsymbol{s}_j(t - \tau_{i,j}). \tag{1.1}$$

In Omlor and Giese (2007) they applied their method on joint angles of recorded emotional full-body motion capture data of the emotions *Happiness, Fear, Sadness*, and *Anger.* They used the angle trajectories of flexions of hip, knee, ankle, spine, clavicle, shoulder, elbow, wrist, neck and head joints.

In the related work Roether et al. (2009) they used their machine learning methods to extract emotion specific features from motion capture data and validated them in experiments with human observers.

The research presented in Barliya et al. (2013) investigates the effects of emotions on the law of intersegmental coordination. They studied the elevation angles of thigh, shank, and ankle and the resulting plane for emotional gaits and gaits of different speeds. Walking velocity still was the largest factor in the change of the orientation of the plane, however changes in the amplitudes of the angles also play an important role when discriminating emotions.

The study presented in Hicheur et al. (2013a) examined how emotions affect postural changes and whether these changes are related to walking velocity. They used two sets of motion capture data. The first was a recording of eight professional actors that expressed *Fear, Sadness, Neutral, Joy*, and *Anger* during walking. The second experiment used five naïve subjects that were asked to walk without expressing emotions but at different speeds (normal, slow, and fast) which the subjects could choose themselves. They compared the two sets of experiments in terms of different parameters, such as walking speed, relative timings of the switch from stance to swing phase, timing of peak amplitudes, and global angles of trunk, head, arm, and forearm. They found that especially the global angles of trunk and head are good indicators for emotion and are much less indicators for different speeds.

The previous findings were used in Hicheur et al. (2013b) to see if and how emotions can be superimposed to animated figures. They first showed that emotions in videos of animated full-light avatars are very well recognized with a recognition rating of $> 85\%$. They then used a neutral motion and modified its speed, the angle of the trunk, the angle of the head, and combinations of all modifications. The changes in speed were $\pm 30\%, \pm 60\%$ and for the angles $\pm 8°, \pm 16°$. Combinations of these changes were used to generate 140 motions which were classified by 32 human observers into emotions *Fear, Sadness, Neutral, Joy, Anger* or *Unknown.* The combinations of of motion parameters with the highest recognition rates are shown in Table 1.1.

In Venture et al. (2014) they investigated whether emotions from recorded emotional walking motions could be automatically recognized with a model with only six or twelve degrees of freedom. They first used inverse kinematics to map motion capture data on a fully articulated 34 degree of freedom model and then used only subsets of the degrees of freedom (DOF). They created two reduced models: the 6DOF model that only used the DOFs of the lower torso and the 12DOF model that adds six additional DOFs for the waist and neck joint (three DOFs each). From the motion capture data they built a database for the two models that allows identifying

Emotion	**Speed**	**Angle Changes**		**Recognition Rate**
		Trunk	Head	
Fear	50%	$0°$	$0°$	50%
Sadness	40%	$-16°$	$-16°$	95%
Joy	130%	$16°$	$0°$	70%
Anger	160%	$-16°$	$0°$	90%

Table 1.1: Recognition rates of artificially superimposed emotion by changing motion parameters as reported in Hicheur et al. (2013b).

emotions by only considering data of these reduced models, however the 12DOF model had in general better recognition rates.

The research of Huis in ‘t Veld et al. (2014) and Huis in ’t Veld et al. (2014a) presents *BACS - the Body Action Coding System* that uses surface electromyograhpic (EMG) data to encode emotion specific patterns in muscular activity. Participants actively expressed or passively viewed bodily expressions of fear while activity of muscles on upper and lower body were recorded. For active expressions they found that bicep, deltoids, forearm, and triceps showed high activity during expression of anger, whereas biceps and deltoids showed high activity during fear. Passive viewing on the other hand showed effects with similar tendencies however on a much smaller scale. Nevertheless it indicates that emotions may already be detected from muscular activity alone.

Chapter 2

Motion Capture Experiments

In this chapter we describe the motion capture experiments that we performed to obtain recordings of full-body emotional expressions. We recorded full-body walking sequences of multiple subjects that express specific emotions. The motion capture experiments were performed at the Laboratoire de la Physiologie et de la Perception de l'Action at the Collège de France in Paris. Additionally to the 3-D marker movement we recorded muscle activity using electromyographic sensors for muscles of upper and lower body to obtain information of muscular activity during the emotional expressions. We evaluated expressiveness of the recordings, i.e. how well the emotions are recognized, by performing an online survey with dynamic point light displays of the recorded motions. Further, we analyze the recorded EMG data for emotion specific patterns.

2.1 Selection of Emotions

As described in Section 1.2.1 there are different approaches to classify emotions by using distinct sets of basic emotions that are used to construct complex emotions. Instead of analyzing a complete set of basic emotions we focus our efforts on two types of emotions, namely *Anger* and *Joy* and compare them to neutral movements. In the following we will refer to neutral movements as expressions of the emotion *Neutral* and use it as a type of emotion in its own.

Focusing on *Anger* and *Joy* is motivated by similar descriptions of the two emotions. Angry or joyful movements are often displayed in film, theater, and animation by increased walking speeds and stronger upper body articulation compared to neutral or sad movements. Various studies show similarities for expressions of *Anger* and *Joy*. For example in Wallbott (1998) both are characterized by the same movement quality terms "expansive" and "high movement dynamic". The analysis performed in Roether et al. (2009) show similar emotion-specific dynamic features for shoulder and elbow flexion.

As with any emotional expression, there is no unique way of expressing an emotion. Rather the contrary holds and there are a wide range of variations for a single emotion. In our work we decided to focus on two specific variants for each *Anger* and *Joy*: one expression that is shown openly and one which is held back or directed inwards. For *Anger* we have *AngerExpressive* and *AngerSupressive* and for *Joy* we have *JoyExpressive* and *JoySelfsufficient*.

2.2 Experiment Overview

Five subjects (one female, four males) participated in this experiment. None of them had any professional acting experience. The motion capture was performed using a Vicon motion capture system with 16 cameras at the Laboratoire de la Physiologie et de la Perception de l'Action at the Collège de France in Paris.

We followed the emotion induction procedure described in Roether et al. (2009). The actual recording was preceded with an induction phase, during which the subjects could freely express the emotion using vocalization, gestures, and facial expressions. There was no time limit for this induction phase, instead the subjects were instructed to go to a specific point in the motion capture lab to indicate to the experimenter that they were ready. The experimenter then started the recording without any interaction with the subject.

The different types of emotion expression were described to the subjects using examples. Subjects were encouraged to alter them or replace them with personal experiences so that they could easier evoke the intended mood. The subjects were instructed to not use any gestures during the recording such as raising and shaking an arm during an *Anger* expression.

AngerExpressive: the motion should be affected by anger and can be expressed openly. *Example:* someone rode with his bike over ones foot, one learned about an affair of ones partner, etc.

AngerSupressive: the motion should be affected by anger, but cannot be expressed openly. *Example:* one was treated injustly in an examination or interview, but an open expression of the anger would make things even worse (failing the exam, lose existing job).

JoyExpressive: the motion should be affected by joy that is directed outwards. *Example:* one won the lottery and goes to see a friend / partner to tell him / her about it (possibly also that one can finally quit the job / buy a house / etc.)

JoySelfsufficient: the motion should be affected by joy that is directed inwards. *Example:* one fully realizes that one has already achieved some of the biggest dreams of ones childhood.

2.2.1 Marker Placement

We recorded the movement using a passive marker-based motion capture system manufactured by Vicon (Vicon, Oxford Metrics Limited, Oxford UK). Reflective markers are attached to the subjects and are then recorded using multiple cameras to reconstruct their spatial positions over time. Markers are usually placed to specific anatomical landmarks (so-called marker sets) to infer the movement of the skeletal system of the recorded subject.

We used the full-body Vicon Plug-In-Gait marker set shown in Figure 2.1. Markers were attached to the subject using adhesive tape directly on the skin to avoid movement artifacts from clothing. Recording, marker labelling, and export to the C3D file format Motion Lab Systems (2008) was done using the Vicon Nexus$^{\text{TM}}$software version 1.3.109.33342. Recording of the marker movement was performed with a frequency of 100 Hz.

2.2.2 Recorded EMG Channels

We recorded muscle activity using surface electromyography (EMG) of muscles on upper and lower body. Surface EMG records electrical signals by measuring electrical signals using electrodes that are placed onto the skin near the muscles.

We used the wireless EMG system Wave Wireless EMG (Cometa, Barregio MI, Italy). Each wireless EMG sensor communicates with a receiver using proprietary wireless protocol. The skin under the electrodes was cleaned with alcohol to minimize impedance before attaching them to the subject. After this we tested the signal of each sensors before proceeding with the next muscle.

In total we recorded 16 muscles that are listed in Table 2.1 along with their associated function. A schematic of the muscles and sensor location is shown in Figure 2.2. The EMG signal was recorded with a frequency of $1000Hz$ using the same software as the marker movement.

2.2.3 Protocol

The experiments were performed in order of the following steps:

1. Brief introduction of the subjects to the research and the recording methods
2. Setup of a new subject model in Vicon Nexus$^{\text{TM}}$

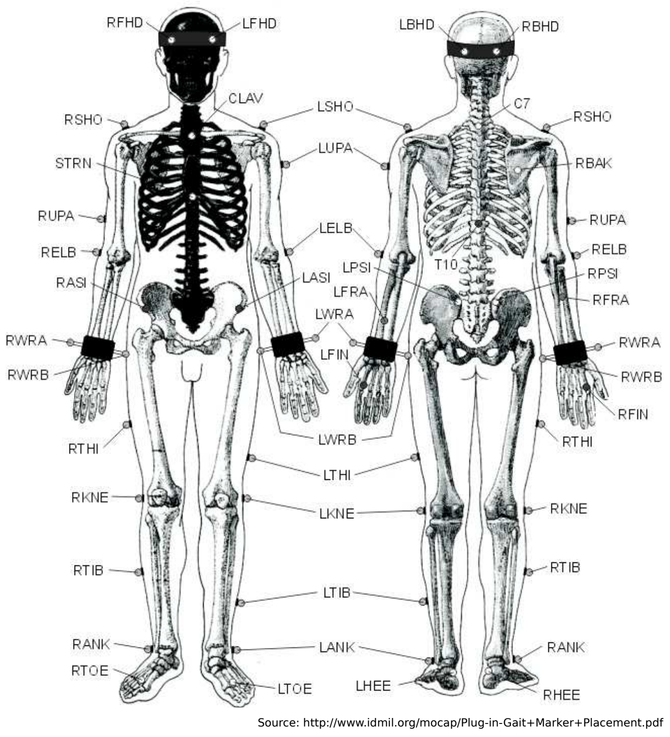


Figure 2.1: Overview of the Plug-In-Gait (PIG) marker placement and labeling.

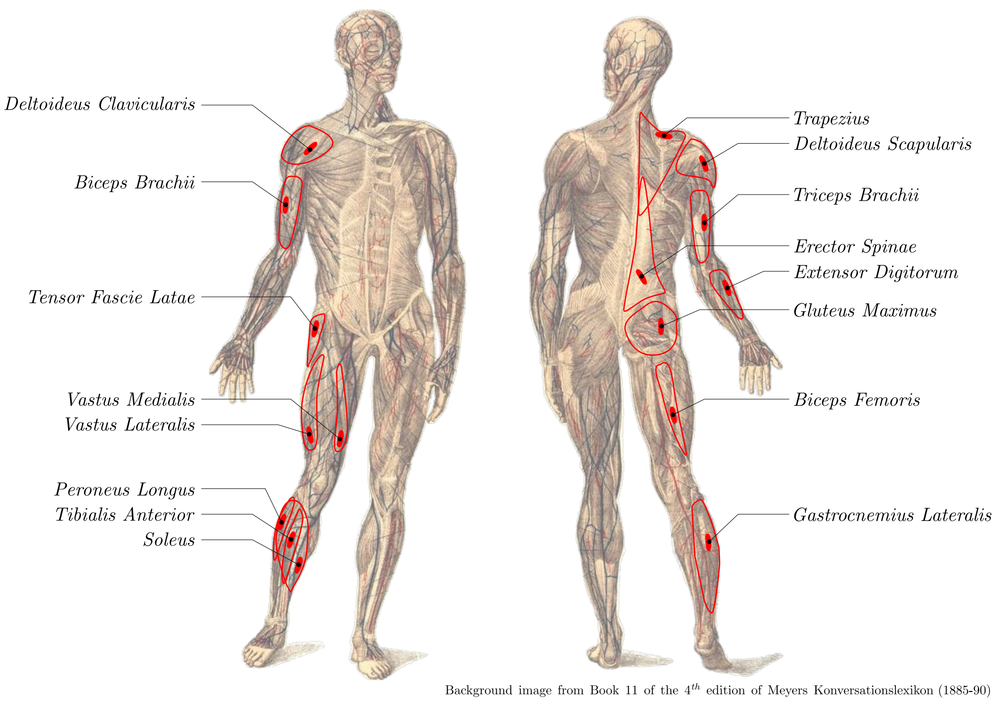


Figure 2.2: Overview of the recorded muscles. The area of the muscle indicated using red lines and the placement of the surface EMG sensors is shown as red dots. All sensors were placed on the right hemisphere of the subject.

Name	Associated Function
Soleus	Plantar flexion of the ankle
Gastrocnemius Lateralis	Plantar flexion of the ankle and flexion of the knee
Tibialis Anterior	Dorsi flexion of the ankle
Peroneus Longus	Plantar flexion and abduction of the ankle
Biceps Femoris	Flexion of the knee
Vastus Lateralis	Extension of the knee
Vastus Medialis	Extension of the knee
Tensor Fascie Latae	Extension of the hip, Abduction of the hip
Gluteus Maximus	Extension of the hip
Erector Spinae	Extension of the back
Trapezius	Abduction of the upper arm and movement of the scapula
Deltoideus Anterior	Flexion and Abduction of the shoulder
Deltoideus Posterior	Extension of the shoulder
Biceps Brachii	Flexion of the elbow
Triceps Brachii	Extension of the elbow
Extensor Digitorum	Extension of the wrist

Table 2.1: Listing of the recorded muscles and their associated main functionality.

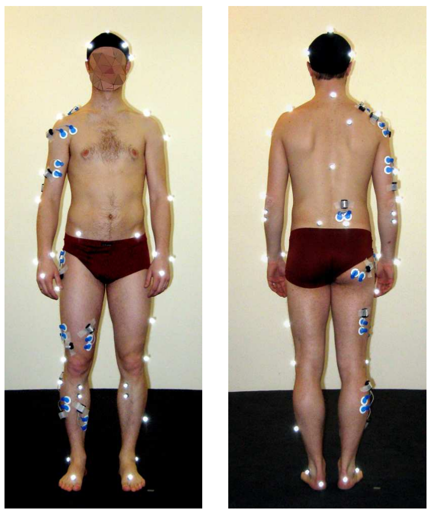
3. Attachment of the EMG sensors to the subject
   (a) Record maximum values for each muscle / maximum cocontraction
4. Attachment of the reflective markers to the subject
5. Recording of five *Neutral* motions
6. Recording of five *AngerExpressive* motions
7. Recording of five *AngerSupressive* motions
8. Recording of five *JoyExpressive* motions
9. Recording of five *JoySelfsufficient* motions
10. Take photos of subject with markers
    (a) T-Pose: from front, back, side left, side right
    (b) Neutral Pose (arms hanging): from front, back, side left, side right

A subject with all motion capture markers and EMG sensors is shown in Figure 2.3.

2.3 Evaluation of the Expressiveness

Once the data was obtained we created short video sequences for each recorded trial. Each video contained a double step sequence. We always used the first double step sequence in a motion capture trial. Unfortunately, some of the trials did not contain all markers for a double step sequence as the motions were very fast. Instead of having 5 trials for each expressed emotion we therefore had 3 trials. This resulted in a total of 75 videos ($75 = 5$ subjects$\times 5$ emotions$\times 3$ trials).

For each double step sequence we created video clips that show the markers from an angle of about 45° and height that is approximate to the subject's head. The camera did not move with the markers and instead is at a fixed position during the whole motion. All markers used the same white color and additionally the ground level was indicated with a grey and black checkerboard pattern with square size $1m$. A sequence of images for one of the used videos is shown in Figure 2.4.



(a) Frontal View (b) Posterior View

Figure 2.3: Frontal and posterior view of a subject on which all motion capture markers and EMG sensors were attached. The markers CLAV and STRN were already removed when the pictures were taken but are present in the motion capture data.

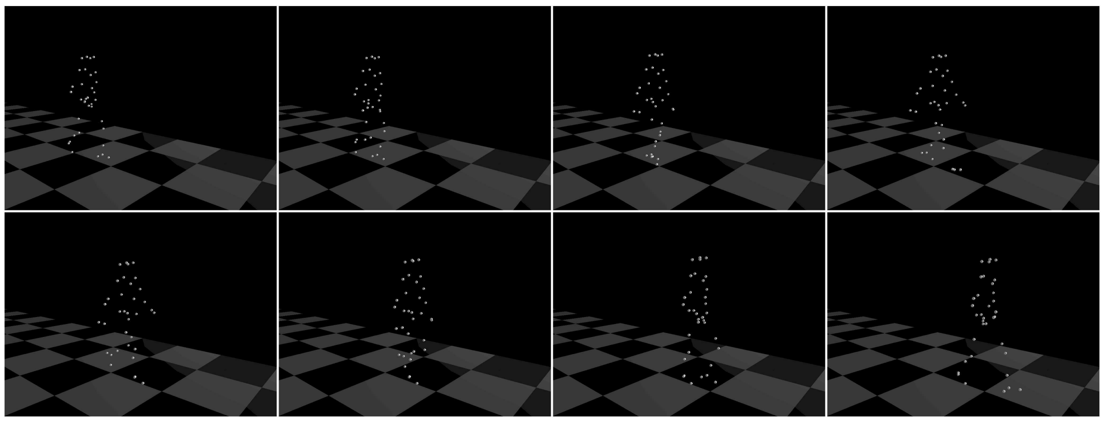


Figure 2.4: Image sequence for one of the double step sequences that were used in the expressiveness evaluation.

2.3.1 Survey Overview

We created an online survey to perform the evaluation. Figure 2.5 shows a screenshot of the webpage which participants used to rate the videos.

In the center it shows the video player with the play button on the top right. The videos did not start automatically and for each click on the play button the video would show just once.

Participants could rate the individual videos on a continuous scale using a slider shown horizontally centered and below the video player. The scale had labels from *Anger* (left) to *Neutral* (center) to *Joy* (right). The participants were instructed to use given extreme values when the motion was perceived as clearly angry (slider to the left) or clearly joyful (slider to the right). Intermediate values should be chosen when there was only a partial agreement with the statements. Furthermore participants could also choose to select "Can't Decide" on a check box right to the scale if the depicted motions did not fit any of the emotions.

On the bottom right the participant could move to the next video by clicking on the button "Next Video". Participants could watch the video as often as wanted but could not return to a previous video to change a previous rating.

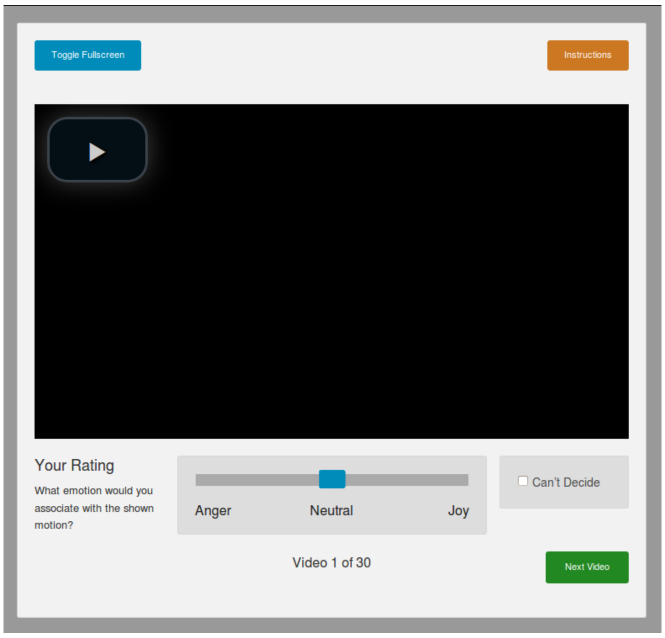


Figure 2.5: Screenshot of the online survey.

The 75 videos were split up in 5 sets of 30 videos each. Each set mainly focused on one subject, i.e. contained all videos of one specific subject and the remaining 15 videos were a mix of 3 additional subjects. The first 5 videos and the last 5 videos used the same subject and the same order of recorded emotions.

Participants could rate one set a time and the survey would choose the set which had been rated the least. The survey furthermore kept track which video was rated last so participants could take a break whenever they wanted and come back later to finish a set. Once a participant rated all sets the survey would notify participants that all sets had been rated and not let rate them further videos.

The survey was completely anonymous and we did not collect any data other than the ratings and at what time those ratings were performed. To reach a high number of possible participants we sent a link to the survey to various mailing lists as well as word-of-mouth (e.g. at conferences) and social media (i.e. Twitter).

2.3.2 Results

There were 109 participants that rated videos in the survey and a total of 4023 ratings were made. On average each video was rated 54.63 ($\pm$3.40) times. The numerical values for mean rating and standard deviation are shown in Table 2.2, a visual representation is shown in Figure 2.6.

The plots in Figure 2.6 show the mean value and standard deviations of the ratings for all videos in the survey. Each plot represents the results for one subject. The y-axis represents the rating where -1 (bottom) represents *Anger*, 0 is *Neutral*, and 1 is *Joy*. The dashed horizontal lines are located at -0.33 and 0.33 which indicate the boundaries of the three emotions on the scale. For each subject the ratings are grouped by the expressed emotions from left to right: *AngerExpressive*, *AngerSupressive*, *Neutral*, *JoySelfsufficient*, and *JoyExpressive*. The three entries in each group are the three trials used in the survey.

For Subject 5 at the bottom center the individual emotions were not recognized by the participants as all mean ratings are within the $(-0.33, +0.33)$ range. The mean ratings and their standard deviations for the expressions for *Neutral* and *JoySelfsufficient* have a relatively small spread and are close to a neutral rating of 0. All other expressions do not display a clear tendency to a specific emotion.

Subject 2 in the top center is similar as for Subject 5: mean ratings are mostly within the

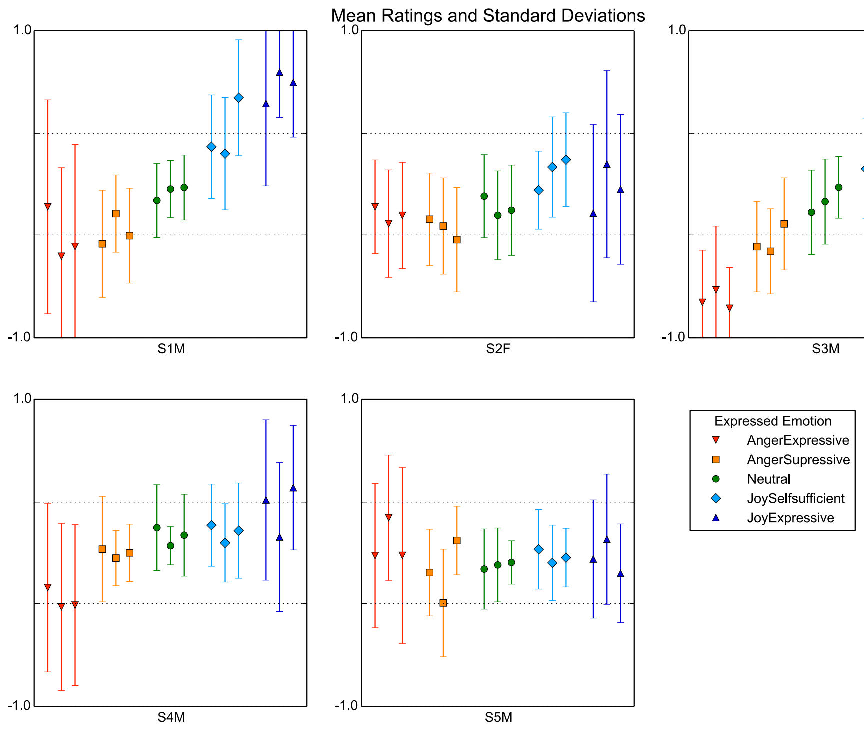


Figure 2.6: Mean rating and standard deviation of the ratings of the double step sequences for all subjects. -1 represents *Anger*, 0 *Neutral*, and 1 corresponds to *Joy*.

$(-0.33, +0.33)$ range, which means that the motions were mostly being interpreted as neutral motions. Two of the 3 motions for *JoySelfsufficient* were interpreted similarly joyful as the most joyful motions for *JoyExpressive*. The ratings for both *AngerSupressive* and *JoySelfsufficient* had clearly lower and higher mean ratings for two videos than all ratings for *Neutral*.

In the bottom row at the left one can see the ratings for Subject 4. One can see a slight tendency of the mean ratings going from -0.33 on the left to $+0.33$ on the right. The second *Neutral* video was recognized very well: its mean value is 0.05 and a standard deviation of ± 0.12. The motions for *AngerExpressive*, while having a large standard deviation, have all their average ratings lower or equal to -0.23. Additionally (taking only the mean values into account) these expressions can be separated from the other emotions. The videos for *JoyExpressive* were similarly well perceived except for the second video, which was perceived as neutral by most.

The results for Subject 1 are shown in the plot on the top left. There is a strong tendency of the mean ratings from bottom left to top right. The motions with the expression *JoyExpressive* have received the highest ratings for all subjects. The second trial has both the highest mean value of 0.73 and the lowest standard deviation of ± 0.29 in this expression for all subjects.

Subject 3 is displayed in the plots at the top right. Of all subjects it shows clear tendency for all expressed emotions except *JoySelfsufficient*, for which the videos were perceived as mostly neutral motions $(0.10 \pm 0.33, 0.00 \pm 0.39, 0.04 \pm 0.34)$. The expressions for *JoyExpressive* were recognized worse than those of Subject 1 but better or equal than all other subjects. The average ratings for the *AngerExpressive* videos have average ratings of lower or equal to -0.69. The third video has both the lowest mean rating of -0.81 and a standard deviation of ± 0.27 which is the lowest standard deviation for all *AngerExpressive* motions across all subjects.

Overall the expressiveness of the five subjects varies greatly. More specifically the expressions of Subjects 2, 4, and 5 were generally difficult to read whereas the videos for Subjects 1 and 3 were recognized very well. For Subject 3 the extreme expressions *JoyExpressive* and *AngerExpressive* span a wider range than those of Subject 1. It is also worth noting that for Subject 3 there is no overlapping of each of the expressions: the closest expressions are *Neutral* (highest average -0.02) and *JoySelfsufficient* (lowest average 0.00).

Both Subject 1 and 3 would be suitable for further analysis, however we chose to use Subject 3, since its ratings and standard deviations were generally similar or better.

	AngerSupressive			**AngerExpressive**		
S1	−0.39±0.35	−0.19±0.25	−0.33±0.31	−0.15±0.70	−0.47±0.58	−0.41±0.66
S2	−0.23±0.30	−0.27±0.31	−0.36±0.34	−0.15±0.30	−0.26±0.35	−0.20±0.35
S3	−0.41±0.29	−0.44±0.28	−0.26±0.30	−0.77±0.34	−0.69±0.42	−0.81±0.27
S4	0.02±0.34	−0.03±0.18	−0.00±0.19	−0.23±0.55	−0.35±0.54	−0.34±0.52
S5	−0.13±0.28	−0.33±0.35	0.08±0.22	−0.02±0.47	0.23±0.41	−0.02±0.57

	JoySelfsufficient			**JoyExpressive**		
S1	0.24±0.34	0.20±0.37	0.56±0.38	0.53±0.54	0.73±0.29	0.66±0.36
S2	−0.04±0.25	0.11±0.33	0.16±0.31	−0.19±0.58	0.13±0.61	−0.03±0.49
S3	0.10±0.33	−0.00±0.39	0.04±0.34	0.48±0.36	0.50±0.38	0.42±0.31
S4	0.18±0.27	0.06±0.25	0.14±0.31	0.34±0.52	0.10±0.49	0.42±0.40
S5	0.02±0.26	−0.07±0.25	−0.03±0.19	−0.04±0.38	0.09±0.42	−0.13±0.32

	Neutral		
S1	−0.11±0.24	−0.03±0.19	−0.02±0.21
S2	−0.08±0.27	−0.20±0.29	−0.17±0.29
S3	−0.18±0.27	−0.11±0.28	−0.02±0.20
S4	0.16±0.28	0.05±0.12	0.12±0.27
S5	−0.11±0.26	−0.08±0.24	−0.06±0.14

Table 2.2: Numerical values for mean rating and standard deviation for all subjects and all videos.

2.4 Analysis of the Electromyographic Data

The recorded raw EMG signal contains high frequencies and, to facilitate analysis various postprocessing methods are commonly employed, as described e.g. in Winter (2009). For our analysis we rectified the raw EMG data filtered using a first order zero-lag Butterworth filter with a cutoff frequency of $6Hz$. When analyzing the data we observed that the muscles *Deltoideus Posterior* and *Biceps Brachii* did not show any signals, so we omitted these in analysis.

In Figure 2.7 we show the EMG data of Subject3 for a double-step sequence. One should note that the data is not normalized with respect to maximum voluntary cocontraction as we here focus on the changes with respect to neutral movements. Each row of plots shows the data for a single muscle and uses the same scale on the vertical axis. Each of the four columns of plots shows the data of one emotional expression together with the EMG data of the neutral motions. All EMG sensors were placed on the right side of the subject and the grey background indicates phases during which the right foot of the subject is in contact with the ground. We only show trials for which data of a complete doublestep sequence was recorded, which were two sequences for *AngerExpressive* and *JoyExpressive* and three for the remaining expressions.

We computed average and standard deviation for each signal and expression. The average signal is shown as a line and the standard deviation as a shaded area.

One can already see that the emotion *AngerExpressive* has the highest levels activation overall. Furthermore, there is a shift of the activation pattern for the *Peroneus Longus* muscle. One of the associated functions of this muscle is the plantar flexion of the foot that propels the body forward. The *Erector Spinae* muscle shows high activity near the end of the of the stance phase and the beginning of the swing phase of the right leg. Another difference to the expression of *Neutral* is that the recording of *Deltoideus Anterior*, which contributes to the flexion of the shoulder, shows increased activity that starts before the touch down of the foot

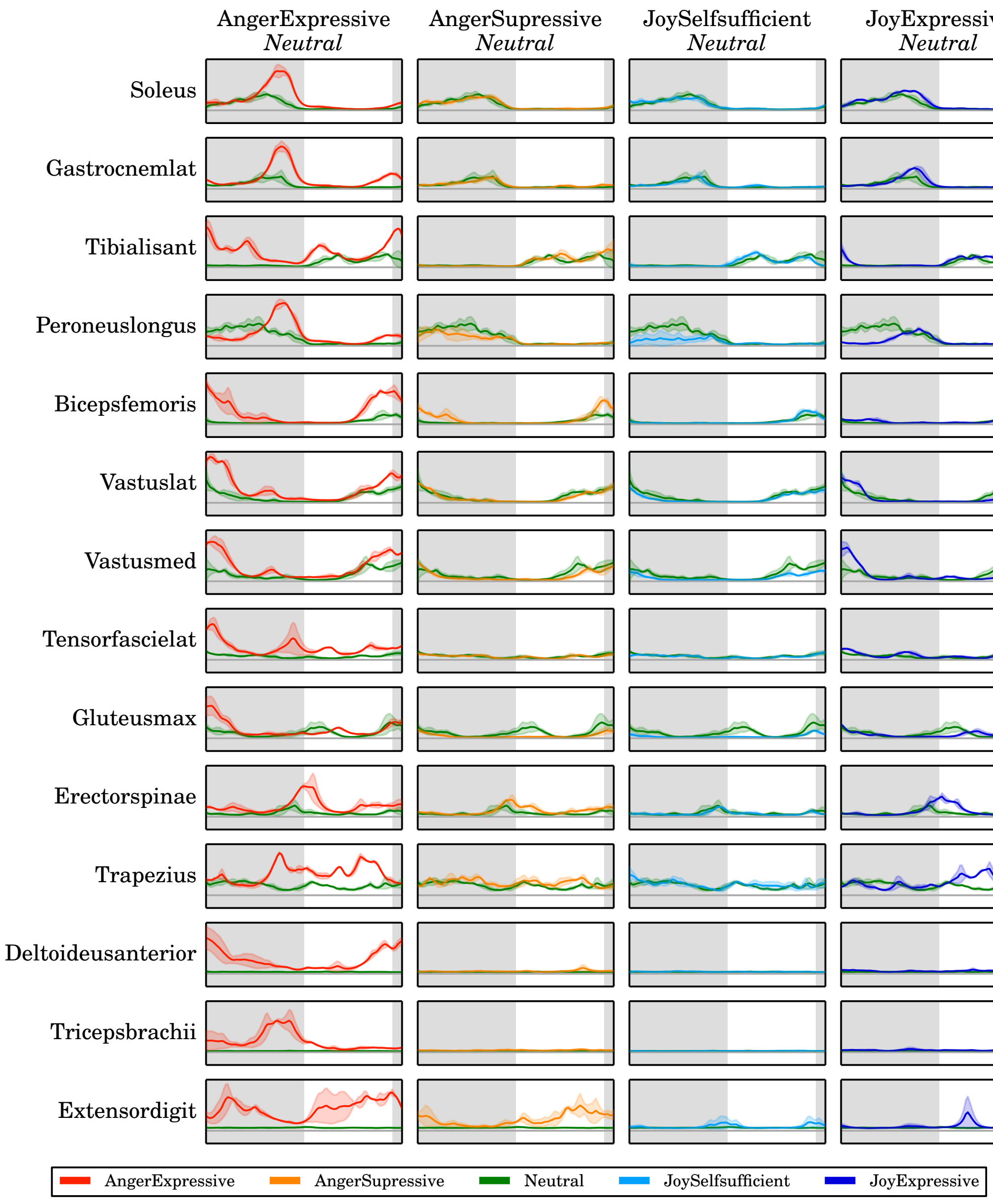


Figure 2.7: Electromyographic data of the recorded muscles of Subject3 for emotion expressions. For each EMG signal and emotional expression we show the average as a line and the standard deviation as an area. Each emotion *AngerExpressive*, *AngerSupressive*, *JoySelfsufficient*, and *JoyExpressive* is shown together with the EMG data of the *Neutral* recordings as a reference. For each muscle the same vertical scale is used. Only muscles of the right side of the subject were recorded and the grey background indicates whether the right foot is in contact with the ground.

and lasts until its lift-off. The measurements of *Triceps Brachii* show activity near the end of the contact phase of the foot. The activity of the *Trapezius*, *Deltoideus Anterior*, and *Triceps Brachii* are likely for the increased arm swinging during the expression of this emotion. The activity shown in *Extensor Digitorum* likely indicates a clenching of the subject's fist. However this remains speculation as neither an antagonist nor enough motion capture markers were used to record this type of movement.

The EMG signals for the expression of *AngerSupressive* is in general very similar to those of

Neutral recordings. The activity of the *Peroneus Longus* is slightly less for this expression than for the *Neutral* recording, however the activity of *Biceps Femoris* shows stronger activity at the early stance phase. Another difference is in the muscle activity of *Extensor Digitorum.* As before, this might be due to clenching of the fist.

For the expression of *JoySelfsufficient* the EMG signals are also very similar to that of *Neutral* expressions and we observe that the *Peroneus Longus* has a lower EMG signal than the *Neutral* trial. For the muscles *Vastus Medialis* and *Gluteus Maximus* the activity is similar or even lower than that of the *Neutral* recordings. This may be related to the shorter average step length and walking velocity compared to the *Neutral* movements. Only for one recorded trial there is higher activity in the *Extensor Digitorum*, which shows some signals in the corresponding plot.

When comparing the muscular activity of *JoyExpressive* with *Neutral* one can see a small temporal shift in the signal of the *Peroneus Longus* and *Erector Spinae.* Also the activity of *Soleus, Gastrocnemius Lateralis*, and *Vastus Medialis* is slightly higher for the expression of *JoyExpressive.* During the swing phase of the right foot one can see an increased activity of *Trapezius.* This muscle contributes to the abduction of the upper arm may therefore support the swinging of the right arm.

It is notable that the activity of *Peroneus Longus* is higher for the expression of *Neutral* compared to those of *AngerSupressive, JoySelfsufficient*, and *JoyExpressive.* This muscle contributes to the plantar flexion and abduction of the foot at the ankle.

Overall one can see clear differences in the EMG data of the different emotional expressions. Especially *AngerExpressive* and *JoyExpressive* show different intensities and timings of peak signals compared to *Neutral* expressions. The generally small standard deviations further support the consistency of our data over multiple trials. Further in-depth analysis and experiments would be required to investigate how much the effects listed above are due to task parameters, such as walking velocity, and also whether the effects can be found across subjects with similar quality of emotional expressiveness.

Part II

Numerical Methods and Models

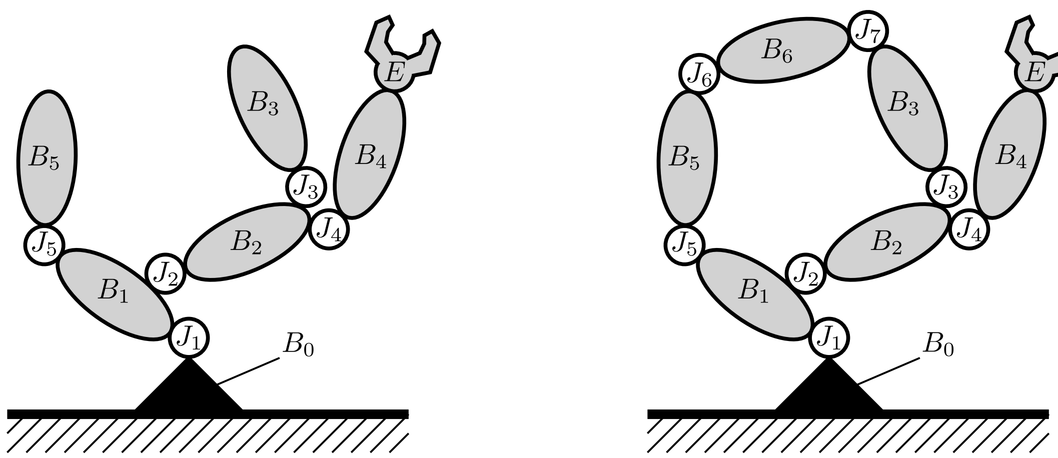
Chapter 3

Efficient Implementation of Rigid Body Dynamics using Spatial Algebra

This chapter presents the theoretical background and implementation details of RBDL - the Rigid-Body Dynamics Library that we created in this thesis to model and compute the dynamics of the rigid-body motions. It is based on Spatial Algebra, a concise notation for rigid-body motions that was established by Roy Featherstone and which is described in detail in his book Featherstone (2008). In large parts we borrow his notation throughout this chapter. Theoretical foundations and algorithms of rigid multibody dynamics have been the subject of various other books, such as Khalil and Dombre (2004); Craig (2005); Shabana (2005); Jain (2011), and others.

3.1 Dynamics of Rigid Multibody Systems

A rigid multibody system is a set of rigid bodies $B_0, B_1, \ldots, B_{n_B}$ that are interconnected by joints $J_1, \ldots, J_{n_J}$. Bodies are rigid in the sense that two points of a single body will always have the same distance. Joints always connect two bodies and induce constraints on the relative motion of the two bodies. There usually is a designated root body from which the whole model is spanned and that is fixed in a global reference frame. We denote this root body as B_0.



(a) Schematic of a rigid multibody system with tree topology

(b) Schematic of a closed-loop system

Figure 3.1: Topologies of rigid multibody systems

Kinematic Trees and Closed-Loop System

The connection of joints and bodies allows us to view the multibody system as a graph where the joints are the edges and the bodies are the nodes (see Figure 3.1). The designated root body (in our case B_0) is then the root node of the graph.

If this graph is a tree then we call the system a *kinematic tree* (Figure 3.1(a)). Various very efficient recursive methods only work on kinematic trees. Otherwise the graph contains loops and the system is called a *closed-loop system* (Figure 3.1(b)). Closed-loop systems can be handled by first considering a sub-system that is a spanning tree and impose loop closure constraints to the system.

Unless stated otherwise the systems in this chapter are assumed to be kinematic trees.

Coordinate Representation

The position and orientation of a single body at a given point in time can be described by 6 variables: 3 translational variables and 3 rotational variables. The state of all bodies in the system can therefore be specified by $6n_B$ variables. However, as the joints limit the relative motion of the connected bodies the actual number of degrees of freedom of the system is in general much smaller. There are two ways of describing the state of a rigid-body system at a given point in time: redundant coordinates or generalized coordinates.

Redundant Coordinates Redundant coordinates (also referred to as "natural coordinates" or "maximal coordinates") use more degrees of freedom than the system actually possesses for the state description. A possible choice of variables would be the $6n_B$ values described earlier. This leads to big structured systems. These structures can be exploited, which results in an increased performance compared to working with the full system. When working with redundant coordinates one also has to ensure that the state variables are consistent with the joints of the system.

Generalized Coordinates Generalized coordinates (also sometimes called "minimal coordinates") describe the state of the system using a vector that has a dimension that is equal to the number of degrees of freedom. An advantage of this approach is that any vector in $\mathbb{R}^{n_{dof}}$ is a valid state description.

In this thesis we use the latter approach, yet the concepts described here are not limited to using generalized coordinates. The generalized state of a rigid-body model is described by $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\tau} \in \mathbb{R}^{n_{dof}}$ where $n_{dof} \in \mathbb{N}$ is the numbers of degrees of freedom of the system. The generalized positions are denoted by $\boldsymbol{q}$, the generalized velocities by $\dot{\boldsymbol{q}}$, the generalized accelerations $\ddot{\boldsymbol{q}}$, and $\boldsymbol{\tau}$ are the generalized forces of the system.

Kinematics and Dynamics

Problem formulations of rigid-body motions can be split up into two areas: kinematics and dynamics. Kinematics is about geometric aspects such as if and how a distinct point on the robot (the so-called *end effector*) can reach a prescribed location in space. Dynamics on the other hand is about the forces that cause the motion and how the system reacts to applied forces or which forces have to act to produce a given motion.

To perform dynamics computations of a multibody system one first has to compute its kinematics. Therefore dynamics problems generally have a higher computational complexity and involve more variables than pure kinematics computations. The focus of this research is on dynamics and the forces that act within the system and also how they change for different inputs. For this we perform a great number of dynamics computations, which require high performance rigid multibody methods.

3.1.1 Equation of Motion for Rigid Multibody Systems

The dynamics of rigid multibody systems using generalized coordinates is described by the following equation:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} \tag{3.1}$$

The matrix $\boldsymbol{H}(\boldsymbol{q})$ is the symmetric and positive definite joint space inertia matrix or generalized inertia matrix. The term $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the generalized bias force and $\boldsymbol{\tau}$ the generalized force applied at the joints. The generalized bias force is equal to the amount of generalized force that has to be applied such that the acceleration $\ddot{\boldsymbol{q}}$ is zero. It contains Coriolis and centrifugal forces, gravity, and other forces acting on the system, which are not caused by $\boldsymbol{\tau}$.

There are two fundamental dynamics problems related to rigid multibody systems: inverse dynamics and forward dynamics. Inverse dynamics computes the generalized forces $\boldsymbol{\tau}$ required to produce given $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$:

$$\boldsymbol{\tau} = \mathit{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{3.2}$$

Forward dynamics computes the acceleration $\ddot{\boldsymbol{q}}$ of the rigid multibody system for given $\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}$. It can be formalized by:

$$\ddot{\boldsymbol{q}} = \mathit{FD}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}). \tag{3.3}$$

Both problems can be directly evaluated using (3.1), yet for sufficiently large n_{dof} much more efficient recursive methods exist.

3.1.2 External Contacts

Equation (3.1) describes the dynamics of a rigid multibody system that is not subject to an external constraint. If one or more points of the model are in contact with the environment then the motion of the model is constrained by these contacts. Additionally the contacts cause forces acting on the model. In this work we only consider holonomic scleronomic constraints which can generally be stated as

$$\boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{0}, \tag{3.4}$$

with $\boldsymbol{g} : \mathbb{R}^{n_{dof}} \to \mathbb{R}^m$, with $m < n_{dof}$ being the number of constraint equations.

For a point constraint that fixes a body point at a specific location the function $\boldsymbol{g}(\boldsymbol{q})$ expresses the difference of the point and its desired position. In this case $m = 3$ where the first entry would be the points X-coordinate, the second its Y-coordinate, and the last its Z-coordinate. If the constraint only acts along one axis for a single point we have $m = 1$, and for a system with n bodies for which every body has one point constraint we have $m = 3n$.

The equation of motion for a rigid multibody model that is subject to an external contact is described by

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} + \boldsymbol{G}(\boldsymbol{q})^T\boldsymbol{\lambda}, \tag{3.5a}$$

$$\boldsymbol{g}(\boldsymbol{q}) = \mathbf{0}, \tag{3.5b}$$

where $\boldsymbol{G}(\boldsymbol{q}) = \frac{\partial}{\partial \boldsymbol{q}}\boldsymbol{g}(\boldsymbol{q})$ is the so-called *contact Jacobian* matrix of size $m \times n_{dof}$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$ are the contact forces. Equation (3.5) is a differential algebraic equation of (differential) index 3. By differentiating the constraint equation (3.5b) twice we obtain:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} + \boldsymbol{G}(\boldsymbol{q})^T\boldsymbol{\lambda}, \tag{3.6a}$$

$$\boldsymbol{G}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{G}}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{0}, \tag{3.6b}$$

which we can rewrite as a linear system of the unknowns $\ddot{\boldsymbol{q}}, \boldsymbol{\lambda}$:

$$\begin{bmatrix} \boldsymbol{H}(\boldsymbol{q}) & \boldsymbol{G}(\boldsymbol{q})^T \\ \boldsymbol{G}(\boldsymbol{q}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\tau} \\ \gamma(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix} \tag{3.7}$$

This system is always solvable if $\boldsymbol{G}(\boldsymbol{q})$ has full rank, which is the case if the constraints in $\boldsymbol{g}(\boldsymbol{q})$ are not redundant. The term $\boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^m$ is the negative right summand of (3.6b) and is also called *contact Hessian*.

To ensure equivalence of (3.5) and (3.7) we have to ensure that the invariants of the constraints are fulfilled:

$$\boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{0}, \tag{3.8}$$

$$\boldsymbol{G}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0}. \tag{3.9}$$

It is sufficient to ensure that these conditions are fulfilled only at the beginning of the contact as due to (3.6b) and therefore (3.7) the constraint conditions are already fulfilled on the acceleration level.

When using numerical integration equation (3.6b) will not be fulfilled exactly, hence errors will accumulate eventually. This is especially true for large step sizes and/or long simulation durations. In this case one can use Baumgarte stabilization (Ascher et al., 1995). In this work however no stabilization was used as the integration horizons are relatively short and no large error accumulations have been observed.

We may omit the arguments $\boldsymbol{q}, \dot{\boldsymbol{q}}$ for the quantities when their requirement should is clear from the context.

3.1.3 Collision Impacts

The transition from a rigid-body system without contacts to a system that has contacts is called a contact gain. During a contact gain very high forces act on the body for a very short time. In the real world these high forces cause the body to first compress and then expand. After the compression phase, depending on the physical properties of the body, the body remains either in contact (perfect inelastic collision) or bounces off.

In rigid-body simulations the compression and expansion is usually neglected and the contact gain is treated as an instantaneous collision. The physical properties of the body are described by the parameter of restitution $e \in [0, 1]$. For $e = 0$ the collision is a perfect inelastic collision, whereas for $e = 1$ the collision is perfectly elastic.

The contact gain is a discontinuous change in the generalized velocity variables from $\dot{\boldsymbol{q}}^-$ to $\dot{\boldsymbol{q}}^+$, i.e. the velocity *before* the collision to the velocity *after* the collision. The change can be computed using

$$\begin{bmatrix} \boldsymbol{H}(\boldsymbol{q}) & \boldsymbol{G}(\boldsymbol{q})^T \\ \boldsymbol{G}(\boldsymbol{q}) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}^+ \\ -\boldsymbol{\Lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{H}(\boldsymbol{q})\dot{\boldsymbol{q}}^- \\ -e\boldsymbol{G}(\boldsymbol{q})\dot{\boldsymbol{q}}^- \end{bmatrix} \tag{3.10}$$

where $\boldsymbol{\Lambda}$ is the contact impulse. The upper part of this equation implies

$$\boldsymbol{H}(\boldsymbol{q})\dot{\boldsymbol{q}}^+ - \boldsymbol{H}(\boldsymbol{q})\dot{\boldsymbol{q}}^- = \boldsymbol{G}(\boldsymbol{q})^T\boldsymbol{\Lambda}, \tag{3.11}$$

which is the change of momentum of the system due to the collision. The lower part

$$\boldsymbol{G}(\boldsymbol{q})\dot{\boldsymbol{q}}^+ = -e\boldsymbol{G}(\boldsymbol{q})\dot{\boldsymbol{q}}^- \tag{3.12}$$

states the contact velocity after the collision. For $e = 0$ this is equivalent to a contact velocity of zero, a perfect inelastic collision.

3.1.4 Modeling Approaches for Reduced Coordinate Rigid Multibody Systems

Manual analytic derivation of the equation of motion is practical for models with very few degrees of freedom. While a planar double pendulum can be derived analytically relatively easy it becomes already arduous to derive a double pendulum with 3-D rotational joints. Adding more bodies and joints with different types of degrees of freedom further increases complexity, which require the use of automatic methods.

There are two common approaches to computing the quantities of the equations of motions (i.e. $\boldsymbol{H}(\boldsymbol{q}), \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, etc.): *symbolic code generation* and *recursive algorithms.* While both essentially compute the same values there are differences in how they are being evaluated and how the modeling is performed. Both approaches take some kind of model description that contains model topology, body and joint specifications as an input to either generate code or to recursively evaluate the quantities.

Symbolic Code Generation

Code generation takes the model description to formulate analytical expressions for the individual quantities in (3.1) and exports the result as computer code. This can be done using any general purpose computer algebra package that allows exporting expressions as code (e.g. MAPLE$^{\text{TM}}$(Maplesoft, 2014).

Using code generation tools allow for optimizations on the symbolic level during the formulation of the expressions and also on the compiler level when the code is transformed into binary code. The code produced by symbolic code generation is very complex and mainly consists of long scalar expressions that are evaluated and stored to the memory locations of the resulting matrices and vectors.

An advantage of this approach is that arbitrary model parameters can easily be treated and also analytic derivatives can be generated. When implemented properly this approach automatically removes expressions that are always 0, e.g. a pure planar model will only evaluate expressions in the plane.

A disadvantage of the code generation approach is that the generation has to be done for every model at least once and also when fundamental changes in the model (e.g. change of coordinates, topology). This generation is an additional step at which may introduce errors by the user and furthermore as the generation process becomes very demanding for the computer algebra package it can also take up several minutes for models as complex as used in this thesis.

Recursive Algorithms

Recursive approaches use the model description as a parameter for a numerical routine to evaluate the individual quantities of the equation of motion. Different models use the same code to compute the quantities. Changes in the model do not require any change in the code and this allows to implement these routines in a library that can easily be run in a host application.

Another advantage of this approach is that it is possible to reason about what is being computed. With symbolic code generation one can only use the quantities it generates code that are used as black box evaluations. Recursive methods on the other hand are described on a higher level of abstraction. That allows improving existing methods and also deriving of new algorithms. It also allow the identification of shared computations that are performed by multiple algorithms. By only evaluating them once one can further enhance the efficiency of these methods.

Recursive methods rely heavily on efficient algorithms and a careful implementation of these. The recursive methods furthermore are described in terms of vector and matrix expressions that can and should be exploited by modern CPU instructions. Therefore efficient linear algebra packages such as LAPACK (Anderson et al., 1999) or EIGEN3 (Guennebaud et al., 2010) are strongly advisable.

3.2 Spatial Algebra

Traditionally the dynamics of rigid bodies are described by two sets of 3-D equations: one 3-D equation for the linear (translational) motions and forces and one 3-D equation for the rotational (angular) motions and forces. Spatial Algebra is a notation for rigid body motions and dynamics that embeds the two types of 3-D equations in a single 6-D equation. A more complete description can be found in Featherstone (2008, 2010a,b).

3.2.1 Elements of Spatial Algebra

The fundamental elements of Spatial Algebra are spatial motions $\hat{\boldsymbol{v}} \in \mathsf{M}^6$, spatial forces $\hat{\boldsymbol{f}} \in \mathsf{F}^6$ and spatial inertias $\hat{\boldsymbol{I}} : \mathsf{M}^6 \rightarrow \mathsf{F}^6$. By using Plücker bases $\mathcal{D}_O, \mathcal{E}_O$ (Featherstone, 2006) we have a mapping $\mathcal{D}_O : \mathbb{R}^6 \rightarrow \mathsf{M}^6$ and $\mathcal{E}_O : \mathbb{R}^6 \rightarrow \mathsf{F}^6$. Also, spatial inertias can then be expressed using 6×6 coordinate matrices with entries in $\mathbb{R}$.

Motion and Force Vectors

A spatial velocity vector describes both the linear and angular velocity of a rigid body and is of the form

$$\hat{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_O \end{bmatrix} = [\omega_x, \omega_y, \omega_z, v_{Ox}, v_{Oy}, v_{Oz}]^T \tag{3.13}$$

where O is an arbitrary but fixed reference point. The 3-D vector $\boldsymbol{\omega} \in \mathbb{R}^3$ describes the angular velocity of the body about an axis that passes through O and $\boldsymbol{v}_O \in \mathbb{R}^3$ is the linear velocity of the body point that currently coincides with the reference point O.

A spatial force vector describes both linear and rotational force components that act on a body and is of the form

$$\hat{\boldsymbol{f}} = \begin{bmatrix} \boldsymbol{n}_O \\ \boldsymbol{f} \end{bmatrix} = [n_{Ox}, n_{Oy}, n_{Oz}, f_x, f_y, f_z]^T \tag{3.14}$$

where $\boldsymbol{n}_O \in \mathbb{R}^3$ describes the total moment about the point that coincides with O and $\boldsymbol{f} \in \mathbb{R}^3$ the linear force along an axis passing through O.

Spatial Inertia and Spatial Momentum

The spatial inertia can be expressed as a 6×6 matrix

$$\hat{\boldsymbol{I}} = \begin{bmatrix} \boldsymbol{I}_C + m\boldsymbol{c}\times\boldsymbol{c}\times^T & m\boldsymbol{c}\times \\ m\boldsymbol{c}\times^T & m\boldsymbol{1} \end{bmatrix} \tag{3.15}$$

where $\boldsymbol{I}_C$ is the inertia of the body at the center of mass, $\boldsymbol{c}$ is the 3-D coordinate vector of the center of mass, and m is the mass of the body. The expression $\boldsymbol{c}\times$ is the skew-symmetric 3×3 matrix that is associated with $\boldsymbol{c}$ and that allows to write the 3-D cross product as a product of a matrix with a vector. It is defined as

$$\boldsymbol{c}\times = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \times = \begin{bmatrix} 0 & -c_z & c_y \\ c_z & 0 & -c_x \\ -c_y & c_x & 0 \end{bmatrix}. \tag{3.16}$$

For a body with spatial velocity $\hat{\boldsymbol{v}}$ and spatial inertia $\hat{\boldsymbol{I}}$ we can compute the spatial momentum as $\hat{\boldsymbol{h}} = \hat{\boldsymbol{I}}\hat{\boldsymbol{v}}$. One should note that $\hat{\boldsymbol{h}}$ is a spatial force vector, i.e. $\hat{\boldsymbol{h}} \in \mathsf{F}^6$.

Transformations

In practice it is useful to use different coordinate frames such as body local coordinate frames instead of performing computations in one single coordinate frame. Spatial Algebra provides a convenient way of transforming spatial quantities such as motions, forces, and inertias between coordinate frames.

If A and B are two coordinate frames where B is translated by $\boldsymbol{r} \in \mathbb{R}^3$ and also rotated as described by the orthonormal matrix $\boldsymbol{E} \in SO(3)$, with $SO(3)$ being the special orthogonal group, we can then define the following spatial transformations

$${}^B\boldsymbol{X}_A = \begin{bmatrix} \boldsymbol{E} & \boldsymbol{0} \\ -\boldsymbol{E}\boldsymbol{r}\times & \boldsymbol{E} \end{bmatrix}, \quad {}^B\boldsymbol{X}_A^* = \begin{bmatrix} \boldsymbol{E} & -\boldsymbol{E}\boldsymbol{r}\times \\ \boldsymbol{0} & \boldsymbol{E} \end{bmatrix} \tag{3.17}$$

with $\boldsymbol{r}\times$ again being the skew-symmetric matrix of the vector $\boldsymbol{r}$.

Elements of M^6 are transformed using ${}^B\boldsymbol{X}_A$ whereas elements of F^6 are transformed using ${}^B\boldsymbol{X}_A^*$. Similarly the inverse transformations ${}^A\boldsymbol{X}_B$ and ${}^A\boldsymbol{X}_B^*$ can be defined. For given $\boldsymbol{E}$ and $\boldsymbol{r}$ we may also write the spatial transformation as $\boldsymbol{X}(\boldsymbol{E}, \boldsymbol{r})$.

The motion vector ${}^A\hat{\boldsymbol{m}} \in \mathsf{M}^6$ and the force vector ${}^A\hat{\boldsymbol{f}} \in \mathsf{F}^6$ that are described in coordinate frame A can be transformed to vectors expressed in the coordinate frame B using

$$ {}^B\hat{\boldsymbol{m}} = {}^B\boldsymbol{X}_A {}^A\hat{\boldsymbol{m}} \tag{3.18} $$

$$ {}^B\hat{\boldsymbol{f}} = {}^B\boldsymbol{X}_A^* {}^A\hat{\boldsymbol{f}}. \tag{3.19} $$

As spatial inertias are mappings from M^6 to F^6 the transformation of spatial inertia ${}^A\hat{\boldsymbol{I}}$ that is expressed in frame A can be transformed to a spatial inertia expressed in frame B by

$$ {}^B\hat{\boldsymbol{I}} = {}^B\boldsymbol{X}_A {}^A\hat{\boldsymbol{I}} {}^A\boldsymbol{X}_B^*. \tag{3.20} $$

Cross Products

When looking at rigid motions using 3-D equations the cross product has a special property: an Euclidean vector $\boldsymbol{r} \in \mathbb{R}^3$ that is fixed in a coordinate frame that has an angular velocity of $\boldsymbol{\omega}$, then the time derivative of $\boldsymbol{r}$ as seen from a non-moving coordinate frame is $\dot{\boldsymbol{r}} = \boldsymbol{\omega} \times \boldsymbol{r}$.

In Spatial Algebra operators with similar properties can be defined. In fact, there are two operators - one for motion vectors:

$$ \hat{\boldsymbol{m}}_O\times = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_O \end{bmatrix} \times = \begin{bmatrix} \boldsymbol{\omega}\times & \boldsymbol{0} \\ \boldsymbol{v}\times & \boldsymbol{\omega}\times \end{bmatrix}, \tag{3.21} $$

and one for force vectors:

$$ \hat{\boldsymbol{v}}\times^* = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_O \end{bmatrix} \times^* = \begin{bmatrix} \boldsymbol{\omega}\times & \boldsymbol{v}_O\times \\ \boldsymbol{0} & \boldsymbol{\omega}\times \end{bmatrix}. \tag{3.22} $$

If the spatial motion vector $\hat{\boldsymbol{m}} \in \mathsf{M}^6$ moves with body i which has a spatial velocity of $\hat{\boldsymbol{v}}_i$, and $\hat{\boldsymbol{m}}$ is constant otherwise, the time derivative of $\hat{\boldsymbol{m}}$ is

$$ \dot{\hat{\boldsymbol{m}}} = \hat{\boldsymbol{v}}_i \times \hat{\boldsymbol{m}}. \tag{3.23} $$

Similarly the time derivative of spatial force vectors in moving reference frames is computed using the $\times^*$ product.

3.2.2 Spatial Equation of Motion

The equation of motion for a single rigid-body expressed using Spatial Algebra is stated by:

$$ \hat{\boldsymbol{f}} = \hat{\boldsymbol{I}}\hat{\boldsymbol{a}} + \hat{\boldsymbol{v}} \times^* \hat{\boldsymbol{I}}\hat{\boldsymbol{v}} \tag{3.24} $$

$$ = \hat{\boldsymbol{I}}\hat{\boldsymbol{a}} + \hat{\boldsymbol{p}}. \tag{3.25} $$

Here $\hat{\boldsymbol{f}} = [\boldsymbol{n}_O, \boldsymbol{f}]^T$ is the spatial force acting on a body where $\boldsymbol{n}_O$ is the total moment applied at point O and $\boldsymbol{f}$ the linear force that is acting on a line passing through O. The spatial acceleration $\hat{\boldsymbol{a}} = [\dot{\boldsymbol{\omega}}, \dot{\boldsymbol{v}}_O]^T$ consists of $\dot{\boldsymbol{\omega}}$ which is the rotational acceleration around an axis passing through O and the linear acceleration $\dot{\boldsymbol{v}}_O$. The quantity $\hat{\boldsymbol{p}}$ is sometimes called the spatial *bias force* which is equal to the spatial force that has to be applied to the body such that no acceleration of the body is produced.

3.2.3 Comparison with the Traditional 3-D Notation

The Spatial Algebra equation of motion of a single rigid-body embeds the traditional 3-D equations of motion by:

$$ \begin{bmatrix} \boldsymbol{n}_C \\ \boldsymbol{f} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_C & \boldsymbol{0} \\ \boldsymbol{0} & m\boldsymbol{1} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \ddot{\boldsymbol{c}} - \boldsymbol{\omega} \times \boldsymbol{v}_C \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega}\times & \boldsymbol{v}_C\times \\ \boldsymbol{0} & \boldsymbol{\omega}\times \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_C & \boldsymbol{0} \\ \boldsymbol{0} & m\boldsymbol{1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_C \end{bmatrix} \tag{3.26} $$

$$ = \begin{bmatrix} \boldsymbol{I}_C\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}_C\boldsymbol{\omega} \\ m\ddot{\boldsymbol{c}} \end{bmatrix}. \tag{3.27} $$

Here the reference point C is assumed to coincide with the center of mass of the body and $\ddot{\boldsymbol{c}}$ is the linear acceleration of the center of mass.

It is important to emphasize that Spatial Algebra is not simply a matter of notation that concatenates the two 3-D equations. Instead it performs operations on a different level of abstraction. Traditional 3-D equations express the motion of rigid bodies using quantities expressed at points that are moving with the body, e.g. the linear velocity of the center of mass and the angular velocity. To compute the velocity or acceleration of the end of a chain one has to transform the quantities from one body to the other, which leads to complicated expressions.

In Spatial Algebra, however, one looks at the flow of the bodies motion at a fixed point in a given reference frame, i.e. the linear and angular velocity of the point of the body that coincides with the reference point at the current instant.

Some of the key advantages of Spatial Algebra are:

- Spatial quantities like velocities, accelerations, and forces behave like proper vectors. This allows one to summing of the velocities of two bodies to compute their combined spatial velocity. Similarly accelerations, forces and inertias can be summed and subtracted.
- Only one 6-D equation instead of two 3-D equations is needed to describe the dynamics. This leads to fewer algebra and also fewer lines of code.
- Rotations, translations and also helical movement are captured by 6-D vectors. This allows one to describe rotational, revolute, and helical joints using the same quantities instead of separate equations that are dependant on the type of joints.
- Concepts can easily be transferred to code. It is possible and also practicable to implement it without loss of efficiency compared to 3-D equations.

3.3 Rigid Multibody Models

In this section we present the rigid multibody model formulation that we implemented in our software RBDL and which we also also use when discussing the algorithms in later sections. It follows the notation and formulation presented in Featherstone (2008).

A rigid multibody system (MBS) consists of a set of bodies that are interconnected by joints. A joints always connects two bodies and in general affects the relative motion of the bodies. The variables that describe a rigid multibody model with $n_B \in \mathbb{N}$ bodies are listed in Table 3.1. It contains variables that describe the structure of the system, namely $\lambda_i, \kappa(i), \mu(i)$, and $\nu(i)$, variables that describe the joints $\boldsymbol{S}_i$, $\boldsymbol{X}_{\mathbf{T}i}$ (motion space and joint location), dynamic properties of the bodies $\boldsymbol{I}_i, \boldsymbol{I}_i^c$, and the current state of the system.

3.3.1 Coordinate Frames and Transformations

There are multiple coordinate frames involved when describing a multibody system. First there is the *global reference frame* which we name 0. This system is the inertial reference frame that is fixed in space. The root body B_0 is attached to this global reference frame and also defines a reference frame that is defined for the body B_0. We call this frame the *body (local) reference frame.* In the case of B_0 the global reference frame and the body reference frame of B_0 always coincide.

When we have a system with a single moving body then this body B_1 is attached via joint J_1 to the base body B_0. Usually joints are not located at the origin of the parent body. Instead they are translated and/or rotated relative to the coordinate frame of the parent body.

This gives rise to the *joint origin frame.* The transformation from the parent body frame to the joint origin frame is denoted as $\boldsymbol{X}_{\mathbf{T}i}$. It is fixed and specified in the parent's body reference frame.

When a joint is moving an additional frame is required. I.e. a revolute joint changes the orientation whereas a prismatic joint results in a translation of the coordinate frame. The *joint motion frame* is the frame that changes when the joint moves and its transformation is denoted as $\boldsymbol{X}_{\mathbf{J}i}$.

Variable	**Description**
λ_i	Index of the parent body index for joint i that connects body i with body λ_i.
$\kappa(i)$	The set of joints that influence (i.e. support) body i.
$\mu(i)$	The set of children of joint i.
$\nu(i)$	The subtree that starts at joint i.
$\boldsymbol{S}_i$	Motion space matrix for joint i.
$\boldsymbol{X}_{\mathbf{T}i}$	Spatial transformation from the parent of body i to the frame of joint i.
$\boldsymbol{v}_i$	Spatial velocity of body i.
$\boldsymbol{a}_i$	Spatial acceleration of body i.
$\boldsymbol{f}_i$	Spatial force that body i exerts onto the parent body λ_i via the connecting joint.
${}^i\boldsymbol{X}_0$	Spatial transformation from the global frame to the frame of body i.
${}^i\boldsymbol{X}_{\lambda_i}$	Spatial transformation from the parent of body i to body i.
$\boldsymbol{I}_i$	Spatial inertia of body i.
$\boldsymbol{I}_i^c$	Composite body inertia of body i.

Table 3.1: Variables of a loop-free rigid multibody model

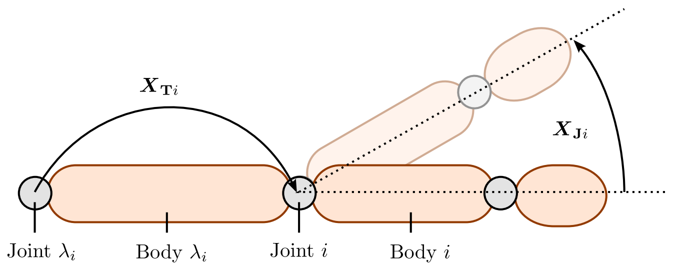

$\boldsymbol{X}_{\mathbf{T}i}$
$\boldsymbol{X}_{\mathbf{J}i}$
Joint λ_i
Body λ_i
Joint i
Body i

Figure 3.2: Schematic of various frame transformations related to articulated rigid body motion. $\boldsymbol{X}_{\mathbf{T}i}$ is the *joint origin frame transformation*, $\boldsymbol{X}_{\mathbf{J}i}$ is the *joint motion frame transformation*. The combined transformation ${}^i\boldsymbol{X}_{\lambda_i} = \boldsymbol{X}_{\mathbf{J}i}\boldsymbol{X}_{\mathbf{T}i}$ is called the *parent frame transformation*.

The transformation from the parent body to the child body is then given by

$${}^i\boldsymbol{X}_{\lambda_i} = \boldsymbol{X}_{\mathbf{J}i}\boldsymbol{X}_{\mathbf{T}i}.$$

Figure 3.2 shows a schematic of the transformations. It is important to note that all transformations $\boldsymbol{X}_{\mathbf{T}i}$, $\boldsymbol{X}_{\mathbf{J}i}$, and ${}^i\boldsymbol{X}_{\lambda_i}$ are full spatial transformations, i.e. they can be rotations, translations, and a combinations of translation and rotation.

3.3.2 Bodies

A single body in a rigid multibody system is described by its mass m, the location of the center of mass $\boldsymbol{c} \in \mathbb{R}^3$ and its inertia tensor $\boldsymbol{I}_C \in \mathbb{R}^{3\times 3}$ at the center of mass. These values are taken to construct the spatial inertia matrix $\hat{\boldsymbol{I}}_i \in \mathbb{R}^{6\times 6}$ for each body i.

3.3.3 Joints

A joint always connects two bodies with each other and in general limits the relative motion between them. A joint may allow between zero and six degrees of freedom for which zero degrees of freedom mean that it is a fixed joint (i.e. the two bodies are rigidly connected with each other) and a joint with 6 degrees of freedom, also called free-flyer joint, does not impose any restriction on the relative motion.

For each joint a specific subset of the generalized state variables are associated with the joint. For joint i with n_s degrees of freedom the value $\boldsymbol{q}_i \in \mathbb{R}^{n_s}$ are the associated generalized joint positions of joint J_i and similarly $\dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i, \boldsymbol{\tau}_i \in \mathbb{R}^{n_s}$ are the associated generalized joint velocities, accelerations and forces for that joint. $\boldsymbol{\tau}_i$ are more specifically the forces that are transmitted via the joint i. The tuple $(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i, \boldsymbol{\tau}_i)$ is referred to as the joint state.

If the generalized joint positions for joint i are nonzero then the transformation caused by the joint is represented by joint motion frame transformation $\boldsymbol{X}_{\mathbf{J}i}$. It is a spatial transformation of the form (3.17) where the entries $\boldsymbol{E}$ and $\boldsymbol{r}$ depend on the type of joint.

Joint Models

The motion space of a joint is described by the so-called *motion subspace matrix* $\boldsymbol{S}$ which defines a mapping $\boldsymbol{S} : \mathbb{R}^{n_s} \rightarrow \mathsf{M}^6$. The subspace matrices for all joints with only one degree of freedom that are either revolute ($\boldsymbol{S}_{R\cdot}$) or prismatic ($\boldsymbol{S}_{T\cdot}$) around or along the coordinate axes are:

$$\boldsymbol{S}_{Rx} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{Ry} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{Rz} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$\boldsymbol{S}_{Tx} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{Ty} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \boldsymbol{S}_{Tz} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

It is also possible to define motion subspace matrices for helical motions. In that case there are nonzero entries in both the upper three and the lower three entries.

Once the motion subspace matrix is defined the joint velocities and accelerations can be expressed as

$$\boldsymbol{v}_{\mathbf{J}i} = \boldsymbol{S}_i \dot{\boldsymbol{q}}_i \tag{3.28}$$

$$\boldsymbol{a}_{\mathbf{J}i} = \dot{\boldsymbol{S}}_i \dot{\boldsymbol{q}}_i + \boldsymbol{S}_i \tag{3.29}$$

$$= \left(\frac{\mathrm{d}\boldsymbol{S}_i}{\mathrm{d}t} + \boldsymbol{v}_i \times \boldsymbol{S}_i \right) \dot{\boldsymbol{q}}_i + \boldsymbol{S}_i \ddot{\boldsymbol{q}}_i \tag{3.30}$$

$$= \boldsymbol{c}_{\mathbf{J}i} + \boldsymbol{v}_i \times \boldsymbol{v}_{\mathbf{J}i} + \boldsymbol{S}_i \ddot{\boldsymbol{q}}_i. \tag{3.31}$$

The first term $\dot{\boldsymbol{S}}_i$ in the acceleration is affected by two factors: *i*) the change of the motion subspace matrix over time and *ii*) a change of the motion subspace matrix due to the movement of the frame of body i in which $\boldsymbol{S}$ is defined. The first factor is nonzero when entries of $\boldsymbol{S}_i$ are not constant but depend on $\boldsymbol{q}_i$. The second factor is a result of (3.23).

The expression $\boldsymbol{c}_{\mathbf{J}i} + \boldsymbol{v}_i \times \boldsymbol{v}_{Ji}$ is called the *velocity dependent spatial acceleration term* as it is independent of the joint acceleration $\ddot{\boldsymbol{q}}_i$. For the simple single degree of freedom joints above, the velocity dependent spatial acceleration term is always zero, however for more complex joints as shown in the following section this is not the case.

Multiple Degrees of Freedom Joints

Models that contain joints with multiple degrees of freedom (DOF) can be treated in two different ways: the emulation of multiple degrees of freedom using multiple single degree of freedom joints and bodies with zero inertia and mass or the use of proper multiple degrees of freedom joints. The former have the advantage that they are very simple to implement however this is at the cost of performance as the algorithms will need more iterations to perform the same computations.

Multiple degrees of freedom joints occur in humanoid robotics mainly in the first movable body of the system from which the remaining multibody system is spanned. In humanoid robotics this body is often referred to as the *Floating Base* and the joint as *Floating Base Joint*, or *Freeflyer Joint*. It has six degrees of freedom and is not actuated, i.e. the generalized forces corresponding to it are always zero. All other joints are usually modeled as 1 DOF joints as actuators with multiple degrees of freedom are difficult to manufacture.

In models for human characters, as used in animations, also other joints such as those of hip, ankle, and shoulder can be modeled using 3 DOF joints. This makes joints with 3 DOF particularly interesting. Furthermore when using 3 DOF joints the floating base can be emulated using two joints instead of six.

For joints with multiple degrees of freedom we need to derive the expressions for $\boldsymbol{E}(\boldsymbol{q}_i), \boldsymbol{r}(\boldsymbol{q}_i)$ to describe the joint motion transformation and $\boldsymbol{S}(\boldsymbol{q}_i)$, and $\boldsymbol{c}_{\mathbf{J}i}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$ to express the motion subspace of the joint, and the velocity dependent spatial acceleration term of the joint.

A 3 DOF joint that describes the translation along the $X-$, $Y-$, and $Z-$ axis is described by

$$\boldsymbol{E}(\boldsymbol{q}_i) = \mathbf{1} \in \mathbb{R}^{3\times 3}, \tag{3.32}$$

$$\boldsymbol{r}(\boldsymbol{q}_i) = (\boldsymbol{q}_{i,0}, \boldsymbol{q}_{i,1}, \boldsymbol{q}_{i,2})^T \in \mathbb{R}^3, \tag{3.33}$$

$$\boldsymbol{S}(\boldsymbol{q}_i) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.34}$$

$$\boldsymbol{c}_{\mathbf{J}i}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) = \mathbf{0} \in \mathsf{M}^6. \tag{3.35}$$

Rotational joints are slightly more involved due to the amount of sine and cosine expressions. For a spherical joint that uses the Euler–Cardan angle convention *YXZ* we use the following notation $s_j = sin(\boldsymbol{q}_{i,j})$ and $c_j = cos(\boldsymbol{q}_{i,j})$ and furthermore $\dot{\boldsymbol{q}}_j = \dot{\boldsymbol{q}}_{i,j}$. This allows us to write the joint model dependent expressions as

$$\boldsymbol{E}(\boldsymbol{q}_i) = \begin{bmatrix} c_2 c_1 & s_2 c_0 + c_2 s_1 s_0 & s_2 s_0 - c_2 s_1 c_0 \\ -s_2 c_1 & c_2 c_0 - s_2 s_1 s_0 & c_2 s_0 + s_2 s_1 c_0 \\ s_1 & -c_1 s_0 & c_1 c_0 \end{bmatrix}, \tag{3.36}$$

$$\boldsymbol{r}(\boldsymbol{q}_i) = \mathbf{0} \in \mathbb{R}^3, \tag{3.37}$$

$$\boldsymbol{S}(\boldsymbol{q}_i) = \begin{bmatrix} c_2 c_1 & s_2 & 0 \\ -s_2 c_1 & c_2 & 0 \\ s_1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{3.38}$$

$$\boldsymbol{c}_{\mathbf{J}i}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) = \begin{bmatrix} -s_2 c_1 \dot{\boldsymbol{q}}_2 \dot{\boldsymbol{q}}_0 - c_2 s_1 \dot{\boldsymbol{q}}_1 \dot{\boldsymbol{q}}_0 + c_2 \dot{\boldsymbol{q}}_2 \dot{\boldsymbol{q}}_1 \\ -c_2 c_1 \dot{\boldsymbol{q}}_2 \dot{\boldsymbol{q}}_0 + s_2 s_1 \dot{\boldsymbol{q}}_1 \dot{\boldsymbol{q}}_0 - s_2 \dot{\boldsymbol{q}}_2 \dot{\boldsymbol{q}}_1 \\ c_1 \dot{\boldsymbol{q}}_1 \dot{\boldsymbol{q}}_0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.39}$$

Joints for other conventions or more degrees of freedom can similarly be derived. To note here is that the presented EulerYXZ joint suffers from singularities. An alternative is a joint that e.g. uses Quaternions (Shoemake, 1985), which is described in Featherstone (2008). For such joints special care has to be taken as the generalized velocities are not the derivatives of the generalized positions anymore.

As suggested in Featherstone (2008) RBDL encapsulates the computations of the joint type and $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ dependent computations in a so-called *joint model calculation routine*:

$$[\boldsymbol{X}_J, \boldsymbol{S}_i, \boldsymbol{c}_{\mathbf{j}i}] = \text{jcalc}(\text{jtype}(i), \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i). \tag{3.40}$$

The method jtype(i) returns a type of joint that will then be used in a **if** ... **elseif** block to choose the actual code depending of the type. This keeps the algorithms independent of joint specific computations and also allows to easily add new joint types.

Joint Numbering

The connection of bodies via joints in a loop-free system can also be seen as a directed graph where the joints are the directed edges and the bodies are the nodes. Apart from the root body there are the same number of bodies B_i as joints J_i with $i = 1, \ldots, n_B$.

Every joint always connects exactly two bodies. Joint i connects body B_{λ_i} with body i. The body with index λ_i is also called the *parent body* and $\lambda_i, i = 1, \ldots, n_B$ is called the *parent array*[1]. The bodies and joints are numbered such that

$$\lambda_i < i \tag{3.41}$$

always holds. If $\kappa(i)$ is the set of joint indices that influence body i then this numbering has the property $j < i,\ \forall j \in \kappa(i)$. The joints with indices in $\kappa(i)$ are also called the *supporting joints* of body i.

This numbering becomes especially useful as various algorithms have a requirement that certain computations must have been performed for the supporting joints before they can be performed for the current joint. With the described joint numbering one can store all joints in an array in the order of their numbering. The mentioned requirement is then already fulfilled when traversing the array from start to end and performing the computation for each joint.

Another important set of indices is $\mu(i)$, which contains the indices of all joints that are contained in the subtree starting at index i.

3.4 Efficient Methods for the Dynamics Computation

In this section we present algorithms that allow us to compute the quantities required for the evaluation of the equation of motion presented in Section 3.1.1. Some of the algorithms are the most efficient algorithms available for their tasks and all algorithms were implemented in the software package RBDL.

3.4.1 Recursive Newton-Euler Algorithm

The Recursive Newton-Euler Algorithm (RNEA) is a highly efficient algorithm for the computation of inverse dynamics of a branched rigid-body model. It has an algorithmic complexity of $O(n_B)$ where n_B is the number of movable bodies in the multibody system.

It performs the computation in three steps:

1. Compute the position, velocity, and acceleration of all bodies.

2. Use for each body (3.25) to compute the net force, which causes the previously computed acceleration.

3. Compute the force that is transmitted over each joint.

[1]One should not confuse the λ of the parent array, i.e. a variable of the rigid body model with the λ that we also use for the contact force. However it should be clear from the context which one is meant.

Pseudo code for the whole algorithm is presented in Algorithm 1. The acceleration of the root body is initialized with $\boldsymbol{a}_g = [\boldsymbol{0}, \boldsymbol{g}]^T \in \mathsf{M}^6$, where $\boldsymbol{g} \in \mathbb{R}^3$ is the gravity vector in global coordinates. As the accelerations are propagated from joint to joint this ensures that gravity is applied to every body.

Algorithm 1: Recursive Newton-Euler Algorithm

```
1  $\boldsymbol{v}_0 = \boldsymbol{0}$
2  $\boldsymbol{a}_0 = -\boldsymbol{a}_g$
3  for $i = 1, \ldots, n_B$ do
4      $[\boldsymbol{X}_J, \boldsymbol{S}_i, \boldsymbol{c}_{\mathbf{J}i}] = \text{jcalc}(\text{jtype}(i), \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$
5      ${}^i\boldsymbol{X}_{\lambda_i} = \boldsymbol{X}_{\mathbf{J}}\boldsymbol{X}_{\mathbf{T}i}$
6      if $\lambda_i \neq 0$ then
7          ${}^i\boldsymbol{X}_0 = {}^i\boldsymbol{X}_{\lambda_i}{}^{\lambda_i}\boldsymbol{X}_0$
8      end
9      $\boldsymbol{v}_{\mathbf{J}i} = \boldsymbol{S}_i\dot{\boldsymbol{q}}_i$
10     $\boldsymbol{a}_{\mathbf{J}i} = \boldsymbol{S}_i\ddot{\boldsymbol{q}}_i + \boldsymbol{c}_{\mathbf{J}i}$
11     $\boldsymbol{v}_i = {}^i\boldsymbol{X}_{\lambda_i}\boldsymbol{v}_{\lambda_i} + \boldsymbol{v}_{\mathbf{J}i}$
12     $\boldsymbol{a}_i = {}^i\boldsymbol{X}_{\lambda_i}\boldsymbol{a}_{\lambda_i} + \boldsymbol{a}_{\mathbf{J}i}$
13     $\boldsymbol{f}_i = \boldsymbol{I}_i\boldsymbol{a}_i + \boldsymbol{v}_i \times^* \boldsymbol{I}_i\boldsymbol{v}_i - {}^i\boldsymbol{X}_0^*\boldsymbol{f}_i^x$
14 end
15 for $i = n_B, \ldots, 1$ do
16     $\boldsymbol{\tau}_i = \boldsymbol{S}_i^T\boldsymbol{f}_i$
17     if $\lambda_i \neq 0$ then
18         $\boldsymbol{f}_{\lambda_i} = \boldsymbol{f}_{\lambda_i} + {}^{\lambda_i}\boldsymbol{X}_i^*\boldsymbol{f}_i$
19     end
20 end
```

The presented algorithm can be applied on kinematic trees and for all scleronomic joints including revolute, prismatic, and helical joints. Furthermore it allows computing the inverse dynamics in presence of external forces $\boldsymbol{f}_i^x \in \mathsf{F}^6$.

The RNEA can be used to compute the generalized bias term $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ of (3.1) by setting $\ddot{\boldsymbol{q}} = \boldsymbol{0}$. However if it is known beforehand that $\boldsymbol{f}_i^x = \boldsymbol{0}, i = 1, \ldots n_B$ a slightly faster variant can be devised by omitting the computation of ${}^0\boldsymbol{X}_i$ and also the product $\boldsymbol{S}_i\ddot{\boldsymbol{q}}_i$ for $i = 1, \ldots, n_B$. This variant is implemented in RBDL in the function `NonlinearEffects (model, q, qdot, C)`.

3.4.2 Composite Rigid-Body Algorithm

The Composite Rigid-Body Algorithm (CRBA) computes the joint space inertia matrix $\boldsymbol{H}(\boldsymbol{q})$. The algorithm proceeds recursively and only computes the entries of the matrix that are nonzero. The algorithm can be motivated by looking at the computation for the multibody system's kinetic energy which can be stated as

$$T = \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{H}\dot{\boldsymbol{q}} = \frac{1}{2}\sum_{i=1}^{n_{dof}}\sum_{j=1}^{n_{dof}} H_{ij}\dot{q}_i\dot{q}_j. \tag{3.42}$$

Alternatively the kinetic energy of the system can be seen as the sum of the kinetic energy of the individual bodies:

$$T = \frac{1}{2}\sum_{i=1}^{n_B}\boldsymbol{v}_i^T\boldsymbol{I}_i\boldsymbol{v}_i. \tag{3.43}$$

Without loss of generality we can assume that all motion subspace matrices are expressed in the global reference frame. This allows us to write (3.43) as

$$T = \frac{1}{2}\sum_{k=1}^{n_B}\sum_{i\in\kappa(k)}\sum_{j\in\kappa(k)}\dot{\boldsymbol{q}}_i^T\boldsymbol{S}_i^T\boldsymbol{I}_k\boldsymbol{S}_j\dot{\boldsymbol{q}}_j. \tag{3.44}$$

This equation expresses the kinetic energy as a sum over all combinations where body k is supported by both the joint i and j. This can be reformulated as

$$T = \frac{1}{2} \sum_{i=1}^{n_B} \sum_{j=1}^{n_B} \sum_{k \in \nu(i) \cap \nu(j)} \dot{\boldsymbol{q}}_i^T \boldsymbol{S}_i^T \boldsymbol{I}_k \boldsymbol{S}_j \dot{\boldsymbol{q}}_j \tag{3.45}$$

$$= \frac{1}{2} \sum_{i=1}^{n_B} \sum_{j=1}^{n_B} \dot{\boldsymbol{q}}_i^T \boldsymbol{H}_{ij} \dot{\boldsymbol{q}}_j \tag{3.46}$$

with

$$\boldsymbol{H}_{ij} = \sum_{k \in \nu(i) \cap \nu(j)} \boldsymbol{S}_i^T \boldsymbol{I}_k \boldsymbol{S}_j. \tag{3.47}$$

Equation (3.44) can be seen as assembling all the required quantities from index k up to the root of the graph whereas equation (3.45) performs the same computation by assembling the quantities down the subtrees starting at the joint that is influenced by both i and j.

The expressions $\boldsymbol{H}_{ij}$ and $\boldsymbol{H}_{ji}$ correspond to the block of the joint space inertia matrix that are affected by the joint variables of joints i and j. Due to the chosen joint numbering we can simplify the union of the two sets using:

$$\nu(i) \cap \nu(j) = \begin{cases} \nu(i) & \text{if } i \in \nu(j) \\ \nu(j) & \text{if } j \in \nu(i) \\ \emptyset & \text{otherwise.} \end{cases} \tag{3.48}$$

Together with

$$\boldsymbol{I}_i^c = \sum_{j \in \nu(i)} \boldsymbol{I}_j \tag{3.49}$$

we can further simplify $\boldsymbol{H}_{ij}$ as

$$\boldsymbol{H}_{ij} = \begin{cases} \boldsymbol{S}_i^T \boldsymbol{I}_i^c \boldsymbol{S}_j & \text{if } i \in \nu(j) \\ \boldsymbol{S}_i^T \boldsymbol{I}_j^c \boldsymbol{S}_j & \text{if } j \in \nu(i) \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{3.50}$$

The Composite Rigid-Body Algorithm is presented as pseudo code in Algorithm 2.

Algorithm 2: Composite Rigid-Body Algorithm

```
1  $\boldsymbol{H} = \boldsymbol{0}$
2  for $i = 1, \ldots, n_B$ do
3      $\boldsymbol{I}_i^c = \boldsymbol{I}_i$
4  end
5  for $i = n_B, \ldots, 1$ do
6      if $\lambda_i \neq 0$ then
7          $\boldsymbol{I}_{\lambda_i}^c = \boldsymbol{I}_{\lambda_i}^c + {}^{\lambda_i}\boldsymbol{X}_i^* \boldsymbol{I}_i^{c\,i} \boldsymbol{X}_{\lambda_i}$
8      end
9      $\boldsymbol{F} = \boldsymbol{I}_i^c \boldsymbol{S}_i$
10     $\boldsymbol{H}_{ii} = \boldsymbol{S}_i^T \boldsymbol{F}$
11     $j = i$
12     while $\lambda_i \neq 0$ do
13         $\boldsymbol{F} = {}^{\lambda_i}\boldsymbol{X}_j^* \boldsymbol{F}$
14         $j = \lambda_i$
15         $\boldsymbol{H}_{ij} = \boldsymbol{F}^T \boldsymbol{S}_j$
16         $\boldsymbol{H}_{ji} = \boldsymbol{H}_{ij}^T$
17     end
18 end
```

The presented algorithm assumes that the spatial transformations from parent to child ${}^i\boldsymbol{X}_{\lambda_i}$ and the joint motion subspace matrices $\boldsymbol{S}_i$ have already been computed. If this was not the case then this could be performed in the first loop. Furthermore the quantities $\boldsymbol{S}_i$ and $\boldsymbol{I}_i^c$ are expressed in body local coordinates.

3.4.3 Articulated Body Algorithm

The Articulated Body Algorithm (ABA) computes the forward dynamics of a kinematic tree with $O(n_B)$ operations. It was first described in Featherstone (1983). However different variants have been derived by others as shown an analyzed in Jain (1991). We only outline the algorithm here. For a complete derivation including some variation in the formulation we refer to Featherstone (2008).

One of its key concepts is the articulated-body inertia which captures the dynamic properties of a body that is connected to other bodies and how it reacts when a force is applied to it while taking the connecting bodies into account. The equation of motion of an articulated body is given by:

$$\boldsymbol{f} = \boldsymbol{I}^A \boldsymbol{a} + \boldsymbol{p}^A \tag{3.51}$$

where $\boldsymbol{I}^A$ is the articulated body inertia and $\boldsymbol{p}^A$ is the articulated bias force.

The articulated body inertia and articulated bias force can be recursively computed. For body i the computations are:

$$\boldsymbol{I}_i^A = \boldsymbol{I}_i + \sum_{j \in \mu(i)} \boldsymbol{I}_j^a, \tag{3.52}$$

$$\boldsymbol{p}_i^A = \boldsymbol{p}_i + \sum_{j \in \mu(i)} \boldsymbol{p}_j^a \tag{3.53}$$

with

$$\boldsymbol{I}_j^a = \boldsymbol{I}_j^A - \boldsymbol{I}_j^A \boldsymbol{S}_j (\boldsymbol{S}_j \boldsymbol{I}_j^A \boldsymbol{S}_j)^{-1} \boldsymbol{S}_j^T \boldsymbol{I}_j^A, \tag{3.54}$$

$$\boldsymbol{p}_j^a = \boldsymbol{p}_j^A + \boldsymbol{I}_j^a \boldsymbol{c}_j + \boldsymbol{I}_j^A \boldsymbol{S}_j (\boldsymbol{S}_j^T \boldsymbol{I}_j^A \boldsymbol{S}_j)^{-1} (\boldsymbol{\tau}_j - \boldsymbol{S}_j^T \boldsymbol{p}_j^A). \tag{3.55}$$

The term $\boldsymbol{c}_j$ is the velocity dependent acceleration term of body i, i.e. $\boldsymbol{c}_j = \boldsymbol{c}_{\mathbf{J}j} + \boldsymbol{v}_j \times \boldsymbol{v}_{\mathbf{J}j}$.

The algorithm proceeds in three steps:

1. Compute positions, velocities and rigid-body bias forces from base outwards.
2. Compute the articulated body inertias and articulated bias forces.
3. From base outwards: use $\boldsymbol{I}_i^A$ and $\boldsymbol{p}_i^A$ computed in the previous step to compute the joint accelerations using:

$$\ddot{\boldsymbol{q}}_i = (\boldsymbol{S}_i^T \boldsymbol{I}_i^A \boldsymbol{S}_i^T)^{-1} (\boldsymbol{\tau}_i - \boldsymbol{S}_i^T (\boldsymbol{I}_i^A \boldsymbol{a}_{\lambda_i} + \boldsymbol{c}_i) - \boldsymbol{S}_i^T \boldsymbol{p}_i^A) \tag{3.56}$$

Pseudocode for the algorithm is shown in Algorithm 3. The code presented here avoids redundant computations of shared terms in Equations (3.54) and (3.55).

The presented algorithm can also efficiently evaluate the forward dynamics for known external forces $\boldsymbol{\lambda}$ and where they act on the model. I.e. it can recursively compute the generalized accelerations $\ddot{\boldsymbol{q}}$ for equations of the form

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} + \boldsymbol{G}(\boldsymbol{q})^T \boldsymbol{\lambda}. \tag{3.57}$$

3.4.4 Point Velocities and Accelerations, Contact Jacobians and Hessians

In simulations and analysis of rigid multibody systems it is often required to compute 3-D coordinates, 3-D linear velocities, and or 3-D linear accelerations of points that are fixed on a

Algorithm 3: Articulated Body Algorithm

```
$\boldsymbol{v}_0 = \boldsymbol{0}$
for $i = 1, \dots, n_B$ do
    $[\boldsymbol{X}_J, \boldsymbol{S}_i, \boldsymbol{c}_{\mathbf{J}i}] = \text{jcalc}(\text{jtype}(i), \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$
    ${}^i\boldsymbol{X}_{\lambda_i} = \boldsymbol{X}_{\mathbf{J}} \boldsymbol{X}_{\mathbf{T}i}$
    if $\lambda_i \neq 0$ then
        ${}^i\boldsymbol{X}_0 = {}^i\boldsymbol{X}_{\lambda_i} {}^{\lambda_i}\boldsymbol{X}_0$
    end
    $\boldsymbol{v}_{\mathbf{J}i} = \boldsymbol{S}_i \dot{\boldsymbol{q}}_i$
    $\boldsymbol{v}_i = {}^i\boldsymbol{X}_{\lambda_i} \boldsymbol{v}_{\lambda_i} + \boldsymbol{v}_{\mathbf{J}i}$
    $\boldsymbol{c}_i = \boldsymbol{c}_{\mathbf{J}i} + \boldsymbol{v}_i \times \boldsymbol{v}_{\mathbf{J}i}$
    $\boldsymbol{I}_i^A = \boldsymbol{I}_i$
    $\boldsymbol{p}_i^A = \boldsymbol{v}_i \times^* \boldsymbol{I}_i \boldsymbol{v}_i - {}^i\boldsymbol{X}_0^* \boldsymbol{f}_i^x$
end
for $i = n_B, \dots, 1$ do
    $\boldsymbol{U}_i = \boldsymbol{I}_i^A \boldsymbol{S}_i$
    $\boldsymbol{D}_i = \boldsymbol{S}_i^T \boldsymbol{U}_i$
    $\boldsymbol{u}_i = \boldsymbol{\tau}_i - \boldsymbol{S}_i^T \boldsymbol{p}_i^A$
    if $\lambda_i \neq 0$ then
        $\boldsymbol{I}^a = \boldsymbol{I}_i^A - \boldsymbol{U}_i \boldsymbol{D}_i^{-1} \boldsymbol{U}_i^T$
        $\boldsymbol{p}^a = \boldsymbol{p}_i^A + \boldsymbol{I}^a \boldsymbol{c}_i + \boldsymbol{U}_i \boldsymbol{D}_i^{-1} \boldsymbol{u}_i$
        $\boldsymbol{I}_{\lambda_i}^A = \boldsymbol{I}_{\lambda_i}^A + {}^{\lambda_i}\boldsymbol{X}_i^* \boldsymbol{I}^a \, {}^i\boldsymbol{X}_{\lambda_i}$
        $\boldsymbol{p}_{\lambda_i}^A = \boldsymbol{p}_{\lambda_i}^A + {}^{\lambda_i}\boldsymbol{X}_i^* \boldsymbol{p}_a$
    end
end
$\boldsymbol{a}_0 = \boldsymbol{0}$ for $i = 1, \dots, n_B$ do
    $\boldsymbol{a}' = {}^i\boldsymbol{X}_{\lambda_i} \boldsymbol{a}_{\lambda_i} + \boldsymbol{c}_i$
    $\ddot{\boldsymbol{q}}_i = \boldsymbol{D}_i^{-1}(\boldsymbol{u}_i - \boldsymbol{U}_i^T \boldsymbol{a}')$
    $\boldsymbol{a}_i = \boldsymbol{a}' + \boldsymbol{S}_i \ddot{\boldsymbol{q}}_i$
end
```

body. This requires some clarifications as spatial motion vectors describe the *flow* of motion at a specified point.

In this section the point P is rigidly attached to (and therefore moving with) the body i. The body local coordinates of P are given by ${}^i\boldsymbol{r}_P \in \mathbb{R}^3$. We are interested in computing the 3-D linear velocity, linear acceleration of the point P in the global reference frame. Furthermore we want to formulate the contact Jacobian and contact Hessian for this point.

There are multiple coordinate frames involved when transforming the spatial quantities to 3-D vectors. These are:

- frame 0: the global coordinate frame,
- frame i: the coordinate frame of body i,
- frame P: the coordinate frame at the point P that is parallel to frame i, and
- frame P': the coordinate frame at the point P that is parallel to the global frame 0.

A schematic of these frames is shown in Figure 3.3.

Point Velocities

The velocity of the fixed body point can be computed using:

$$ {}^{P'}\hat{\boldsymbol{v}}_i = \begin{bmatrix} {}^{P'}\boldsymbol{\omega} \\ {}^{P'}\boldsymbol{v}_{P'} \end{bmatrix}_i = \boldsymbol{X}({}^0\boldsymbol{E}_i^T, {}^i\boldsymbol{r}_P){}^i\boldsymbol{v}_i \tag{3.58} $$

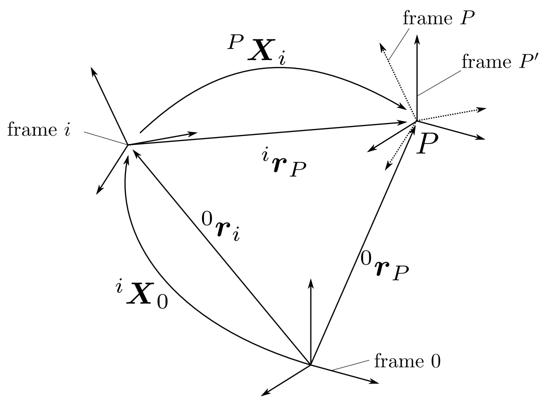


Figure 3.3: Coordinate frames involved in computing conventional 3-D quantities. Frame 0 is the global coordinate frame, frame i is the coordinate of body i, frame P is a coordinate frame at the point P that is parallel to frame i, and frame P' is a coordinate frame at point P that is parallel to the global coordinate frame.

where ${}^0\boldsymbol{E}_i$ is the orientation of body i with respect to the orientation of the global frame. The spatial motion vector $\boldsymbol{v}_i$ is transformed to the coordinate frame P'. The lower part of ${}^{P'}\hat{\boldsymbol{v}}_P$ is therefore expressed at the point of body i that currently coincides with P and therefore we have ${}^{P'}\boldsymbol{v}_{P'} = {}^0\dot{\boldsymbol{r}}_P$.

In RBDL the function `CalcPointVelocity (model, q, qdot, body_id, point_position)` performs this computation and returns the 3-D vector ${}^{P'}\boldsymbol{v}_{P'}$.

Point Accelerations

To compute the acceleration $\ddot{\boldsymbol{r}}_P$ of the point P with fixed body coordinates we can use:

$$ {}^{P'}\hat{\boldsymbol{a}}_i = \begin{bmatrix} {}^{P'}\boldsymbol{\omega} \\ {}^{P'}\boldsymbol{a}_{P'} \end{bmatrix}_i = \boldsymbol{X}({}^0\boldsymbol{E}_i^T, {}^i\boldsymbol{r}_P){}^i\boldsymbol{a}_i + \begin{bmatrix} \boldsymbol{0} \\ {}^{P'}\boldsymbol{\omega} \times {}^{P'}\boldsymbol{v}_{P'} \end{bmatrix}. \tag{3.59} $$

The first term on the right-hand side transforms the spatial acceleration from body i coordinates to the coordinate frame of P' (i.e. the same coordinate frame as above). The second term compensates for the fact that spatial acceleration is *not* the acceleration of the reference point of the coordinate frame but instead it is *the rate of change of flow* at the reference point of the coordinate system!

In RBDL the function `CalcPointAcceleration (model, q, qdot, qddot, body_id, point_position)` performs this computition and returns the 3-D vector ${}^{P'}\boldsymbol{a}_{P'}$.

Contact Jacobians

For a given joint configuration $\boldsymbol{q}$ the contact Jacobian from (3.5) maps from the generalized velocities $\dot{\boldsymbol{q}}$ to the spatial velocity of a body expressed in some coordinate frame A:

$$ {}^A\boldsymbol{v}_j = {}^A\hat{\boldsymbol{G}}(\boldsymbol{q})\dot{\boldsymbol{q}} = \begin{bmatrix} {}^A\boldsymbol{G}_\omega(\boldsymbol{q}) \\ {}^A\boldsymbol{G}_{v_A}(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} \tag{3.60} $$

${}^A\hat{\boldsymbol{G}}(\boldsymbol{q})$ is a $6 \times n_{dof}$ matrix where the upper three rows ${}^A\boldsymbol{G}_\omega(\boldsymbol{q})$ map onto the angular velocity of the body and the bottom three rows ${}^A\boldsymbol{G}_{v_A}(\boldsymbol{q})$ map onto the linear velocity of ${}^A\boldsymbol{v}_j$. All columns with index $i \notin \kappa(j)$ are zero. The values of the non-zero columns can be obtained by rewriting (3.60) as

$$ {}^A\hat{\boldsymbol{G}}(\boldsymbol{q})\dot{\boldsymbol{q}} = \sum_{i\in\kappa(j)} {}^A\boldsymbol{X}_i\boldsymbol{v}_i = \sum_{i\in\kappa(j)} {}^A\boldsymbol{X}_i\boldsymbol{S}_i\dot{q}_i. \tag{3.61} $$

The nonzero columns of the Jacobian can therefore be be obtained by transforming the joint motion space matrices to the coordinate frame A. By using the coordinate frame P' from the previous paragraphs one can compute the Jacobian for the point P that is fixed in body i.

Contact Hessians

When differentiating the algebraic constraint equation twice we obtain

$$\begin{aligned}\frac{\mathrm{d}^2}{\mathrm{d}t^2}\boldsymbol{g}(\boldsymbol{q}) &= \boldsymbol{G}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{G}}(\boldsymbol{q})\dot{\boldsymbol{q}} \qquad (3.62)\\ &= \boldsymbol{G}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \gamma(\boldsymbol{q},\dot{\boldsymbol{q}}), \qquad (3.63)\end{aligned}$$

where $\boldsymbol{\gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is also called the contact Hessian. In RBDL it is computed using the same equations as the point acceleration equations (3.59) and setting $\ddot{\boldsymbol{q}} = \boldsymbol{0}$.

3.4.5 Center of Mass, and Angular Momentum Computation

Using spatial algebra we can also compute the system's spatial inertia which is the inertia the system would have if we lumped all bodies to a single body. It is obtained using:

$$^0\boldsymbol{I} = \sum_{i=1}^{n_B} {}^0\boldsymbol{X}_i^* \boldsymbol{I}_i \; {}^i\boldsymbol{X}_0. \qquad (3.64)$$

The diagonal elements in lower right 3×3 submatrix of $^0\boldsymbol{I}$ all contain the value of the system's total mass. The lower left 3×3 matrix contains the expression $m\boldsymbol{c}\times^T$, where $\boldsymbol{c}$ is the location of the system's Center of Mass (CoM) for which the 3-D vector can easily be extracted.

For a single rigid body its spatial momentum is $\boldsymbol{h} = \boldsymbol{I}\boldsymbol{v}$, which is an element of F^6. Together with the spatial transformations we can write the system's spatial momentum expressed at the origin of the global reference frame as

$$^0\boldsymbol{h} = \sum_{i=1}^{n_B} {}^0\boldsymbol{X}_i^* \boldsymbol{I}_i \boldsymbol{v}_i. \qquad (3.65)$$

The linear part of this vector will contain the linear momentum of the system which allows us to obtain the linear velocity of the CoM by dividing it by the system's total mass. The upper part contains the system's angular momentum expressed at the origin of the global coordinate system. Using the CoM's location we can get the Centroidal momentum Orin and Goswami (2008) as the upper 3-D vector of

$$^{CoM}\boldsymbol{X}_0 {}^0\boldsymbol{h}. \qquad (3.66)$$

3.4.6 Numerical Solution of the Contact Dynamics

There are different ways of solving the linear system of the equation of motion with external contacts (3.7). By setting $\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{\tau}) = -\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{\tau}$ and omitting the arguments $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\boldsymbol{\tau}$ we can write the system as

$$\begin{bmatrix} \boldsymbol{H} & \boldsymbol{G}^T \\ \boldsymbol{G} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \ddot{q} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c} \\ \gamma \end{bmatrix}, \qquad (3.67)$$

where $\boldsymbol{H} \in \mathbb{R}^{n_{dof} \times n_{dof}}$, $\boldsymbol{G} \in \mathbb{R}^{m \times n_{dof}}$, and with $0 < m < n_{dof}$ being the number of contact constraints.

The matrix in the linear system is of the same structure as the so-called *Karush-Kuhn-Tucker* (KKT) matrix that plays an important role in quadratic optimization problems. The matrix is regular if $\boldsymbol{G}$ has full rank and is furthermore indefinite (Nocedal and Wright, 2006).

Direct Solution

The matrix in (3.67) is symmetric but indefinite. An efficient method to solve the system is the symmetric indefinite LDL^T factorization. Nevertheless more efficient methods are available that exploit the structure of the matrix.

Range-Space Method

The so-called range-space method (or sometimes called *Schur-complement method*) first solves for $\boldsymbol{\lambda}$ and then $\ddot{\boldsymbol{q}}$. The former is computed using

$$\boldsymbol{G}\boldsymbol{H}^{-1}\boldsymbol{G}^T\boldsymbol{\lambda} = \boldsymbol{\gamma} - \boldsymbol{G}\boldsymbol{H}^{-1}\boldsymbol{c} \tag{3.68}$$

and the latter is obtained by solving for $\ddot{\boldsymbol{q}}$ using the system

$$\boldsymbol{H}\ddot{\boldsymbol{q}} = \boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda}. \tag{3.69}$$

(3.68) can be obtained by multiplying the upper part of (3.67) with $\boldsymbol{G}\boldsymbol{H}^{-1}$ and then subtracting the result from the bottom part.

The matrices $\boldsymbol{G}\boldsymbol{H}^{-1}\boldsymbol{G}^T$ and $\boldsymbol{H}$ are both symmetric and positive definite which allows us to use a Cholesky decomposition to solve the equations (3.68) and (3.69). For further improvements it suffices to factorize $\boldsymbol{H}$ just once and use the resulting factors to compute the columns of $\boldsymbol{H}^{-1}\boldsymbol{G}^T$ and the vector $\boldsymbol{H}^{-1}\boldsymbol{c}$ in (3.68) and furthermore for (3.69) to compute the accelerations. Decompositions that exploit sparsities in the joint space inertia matrix are described in Featherstone (2005) and Featherstone (2008).

Another way to compute the linear system (3.68) is described in Kokkevis and Metaxas (1998). Instead of explicitly computing and factorizing $\boldsymbol{H}$ it recursively computes the matrix $\boldsymbol{G}\boldsymbol{H}^{-1}\boldsymbol{G}^T$ by applying a so-called test-force for each constraint and compute the resulting accelerations for all constraints. This information is then used to build the matrix column by column. This method requires the application of a test-force and computation of the constraint acceleration to be performed efficiently. For the former one can use a modified version of the Articulated Body Algorithm which only evaluates variables that change due to the test forces. E.g. one should avoid recomputing quantities like ${}^{i}\boldsymbol{X}_{\lambda i}$ or $\boldsymbol{v}_i$. Due to the high cost of the Articulated Body Algorithm (even the modified variant) this approach is only advisable when very few constraints are posed on the system.

Null-Space Method

The null-space method first computes joint accelerations $\ddot{\boldsymbol{q}}$ and then (if necessary) the contact forces $\boldsymbol{\lambda}$. Let $\boldsymbol{D} = [\,\boldsymbol{Y} \mid \boldsymbol{Z}\,] \in \mathbb{R}^{n\times n}$ be a given non-singular matrix with $\boldsymbol{Y} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{Z} \in \mathbb{R}^{n\times(n-m)}$ and $\boldsymbol{G}\boldsymbol{Z} = \boldsymbol{0}$. As both $\boldsymbol{D}$ and $\boldsymbol{G}$ are full-rank we can follow that the $m \times m$ matrix $\boldsymbol{G}\boldsymbol{Y}$ is also non-singular. Furthermore using $\boldsymbol{Y}$ and $\boldsymbol{Z}$ we can partition $\ddot{\boldsymbol{q}}$ as

$$\ddot{\boldsymbol{q}} = \boldsymbol{Y}\ddot{\boldsymbol{q}}_Y + \boldsymbol{Z}\ddot{\boldsymbol{q}}_Z \tag{3.70}$$

with $\ddot{\boldsymbol{q}}_Y \in \mathbb{R}^m$ and $\ddot{\boldsymbol{q}}_Z \in \mathbb{R}^{n-m}$.

Inserting (3.70) into the lower part of (3.67) allows us to compute $\ddot{\boldsymbol{q}}_Y$ by solving

$$\boldsymbol{G}\boldsymbol{Y}\ddot{\boldsymbol{q}}_Y = \boldsymbol{\gamma}. \tag{3.71}$$

To compute $\ddot{\boldsymbol{q}}_Z$ we insert (3.70) into the upper part of (3.67), which gives us

$$\boldsymbol{H}(\boldsymbol{Y}\ddot{\boldsymbol{q}}_Y + \boldsymbol{Z}\ddot{\boldsymbol{q}}_Z) - \boldsymbol{G}^T\boldsymbol{\lambda} = \boldsymbol{c}. \tag{3.72}$$

Premultiplying this with $\boldsymbol{Z}^T$ allows us to compute $\ddot{\boldsymbol{q}}_Z$ using

$$\boldsymbol{Z}^T\boldsymbol{H}\boldsymbol{Z}\ddot{\boldsymbol{q}}_Z = \boldsymbol{Z}^T(\boldsymbol{c} - \boldsymbol{H}\boldsymbol{Y}\ddot{\boldsymbol{q}}_Y). \tag{3.73}$$

The contact forces can then be computed using the upper part of (3.67). By premultiplying it with $\boldsymbol{Y}^T$ and rearranging it we obtain

$$(\boldsymbol{G}\boldsymbol{Y})^T\boldsymbol{\lambda} = \boldsymbol{Y}^T(\boldsymbol{H}\ddot{\boldsymbol{q}} - \boldsymbol{c}), \tag{3.74}$$

which can be solved for $\boldsymbol{\lambda}$.

The matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$ can be computed using a QR decomposition of $\boldsymbol{G}^T$, i.e.

$$\boldsymbol{G}^T \boldsymbol{P} = [\boldsymbol{Q}_1 \mid \boldsymbol{Q}_2] \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}, \tag{3.75}$$

with the permutation matrix $\boldsymbol{P} \in \mathbb{R}^{m \times m}$ and the matrices $\boldsymbol{Y} = \boldsymbol{Q}_1$ and $\boldsymbol{Z} = \boldsymbol{Q}_2$, which fulfill the requirements for $\boldsymbol{Y}$ and $\boldsymbol{Z}$.

One advantage of the null-space method is that one can compute the accelerations separately and compute the contact forces only when required. E.g. a pure simulator that does not make use of the contact forces may take advantage of this slight improvement.

However in general it is not clear whether the range-space method or null-space method is best suited to solve the contact system. For very small m it even may be faster to just solve the full system directly. The topology of the rigid multibody system, the number of constraints, but also which bodies of the system are constrained affect sparsity patterns in the matrices $\boldsymbol{H}$ and $\boldsymbol{G}$, which may be possible to exploit.

3.5 Implementation Details to RBDL

RBDL was implemented in C++ and is publicly available at `http://rbdl.bitbucket.org` along with its documentation. It is published under the permissive zlib license which allows use of modified versions in commercial closed-source projects. The code is extensively tested using more than 180 automated tests.

3.5.1 Model Structure and Data Layout

A central component of RBDL is the model structure `Model`. It contains all the variables from Table 3.1 plus additional variables such as the total number of degrees of freedom, optional names for the bodies, the direction of gravity and further variables for bookkeeping.

The individual variables of Table 3.1 are stored contiguously in arrays, i.e. all the spatial velocities of all bodies are stored in `Model.v` such that `Model.v[i]` holds the value of $\boldsymbol{v}_i$, the spatial velocity of body i.

3.5.2 Programming Interface Considerations

RBDL uses a strict subset of C++ that avoids inheritance and the use of virtual functions as both have an impact on readability and performance due to virtual method table lookups.

Furthermore by using structs instead of classes (i.e. only public member variables) we allow direct access to all the model variables. This can be helpful when debugging, but more importantly it simplifies the creation of automated tests.

One of the aims of the programming interface was to closely follow the notation used here and in Featherstone (2008). This facilitates understanding of the code and also simplifies implementing of the algorithms.

Functions and their arguments closely follow mathematical concepts. E.g. to compute the forward dynamics (i.e. $\ddot{\boldsymbol{q}}$) for a given model and given $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\boldsymbol{\tau}$ one has to call:

```
ForwardDynamicsLagrangian (model, q, qdot, tau, qddot);
```

where `model` is the Model structure, `q` is $\boldsymbol{q}$, `qdot` is $\dot{\boldsymbol{q}}$, `tau` is $\boldsymbol{\tau}$, and `qddot` is $\ddot{\boldsymbol{q}}$. The function builds and solves (3.1) for $\ddot{\boldsymbol{q}}$. All arguments are passed by reference to avoid unnecessary copying including the last argument which is used to return the computed values.

Similarly inverse dynamics is computed using

```
InverseDynamics (model, q, qdot, qddot, tau);
```

which performs the computation of the Recursive Newton-Euler Algorithm listed in Algorithm 1. Again the result is returned using the last argument `tau`.

An example for the RBDL programming interface is shown in Algorithm 4. It creates a 3-D triple pendulum and computes the forward dynamics using the Articulated Body Algorithm and prints out the generalized acceleration.

Algorithm 4: A complete RBDL example of the modeling and forward dynamics computation of a triple pendulum with spherical joints that are parameterized with EulerZYX angles using 3 DOF joints.

```
#include "rbdl.h"
#include <iostream>

using namespace RigidBodyDynamics;
using namespace RigidBodyDynamics::Math;

int main (int argc, char* argv[]) {
  Model model;
  Joint joint_rot_zyx ( JointTypeEulerZYX );
  Body body (0.1, Vector3d (0., 0., -1.), Matrix3d (
    0.1, 0., 0.,
    0., 0.1, 0.,
    0., 0., 0.1)
  );

  model.gravity = Vector3d (0., 0., -9.81);
  unsigned int body_1_id = model.AddBody (
    0,
    Xtrans (Vector3d ( 0., 0., 0.)),
    joint_rot_zyx,
    body);
  unsigned int body_2_id = model.AddBody (
    body_1_id,
    Xtrans (Vector3d ( 0., 0., -1.)),
    joint_rot_zyx,
    body);
  unsigned int body_3_id = model.AddBody (
    body_2_id,
    Xtrans (Vector3d ( 0., 0., -1.)),
    joint_rot_zyx,
    body);

  VectorNd q = VectorNd::Zero (model.q_size);
  VectorNd qdot = VectorNd::Zero (model.qdot_size);
  VectorNd qddot = VectorNd::Zero (model.qdot_size);
  VectorNd tau = VectorNd::Zero (model.tau_size);

  q[0] = 1.2;
  ForwardDynamics (model, q, qdot, tau, qddot);

  std::cout « qddot.transpose() « std::endl;

  return 0;
}
```

3.5.3 Constraint Sets

Another important data structure of RBDL is `ConstraintSet`. It contains all the information needed to compute the dynamics of the model when it is subject to external constraints and also collisions. Additionally it contains workspace memory for temporary quantities that are used when performing the computations. It furthermore stores computed values such as contact forces and collision impulses.

When creating a ConstraintSet it is empty by default. Constraints are added by calling `ConstraintSet::AddConstraint(...)` and specifying the index of the body on which the constraint should act, the location of the point of the body at which the constraint force acts (specified in body-local coordinates), and the 3-D normal in base coordinates along which the constraint acts.

For example a point contact for which the translation of a point is constrained would be modeled using three calls to `ConstraintSet::AddConstraint(...)` with three different normals and otherwise the same parameters for each call.

The workspaces are by default not initialized. Once all constraints have been added one has to call `ConstraintSet::Bind(model)` which then allocates the workspaces.

Once the constraint set has been bound one can compute the forward dynamics of a system that is constrained by the constraint set `CS` is done using:

```
ForwardDynamicsContactsDirect (model, q, qdot, tau, CS, qddot);
```

which will build and solve system (3.7). The computed accelerations will be stored in `qddot` and the computed contact forces in `CS.lambda` which has as many entries as there are constraints. Similarly RBDL provides methods `ForwardDynamicsContactsRangeSpaceSparse(...)` and `ForwardDynamicsContactsNullSpace(...)`.

An example code is shown in Listing 5. Here we add three constraints to a body. The constraints all act on the same body point and constrain the motion of the point in all cardinal directions. After this we bind the constraint set to the model. This causes the workspace to be allocated and must be performed before computing the contact dynamics by calling `ForwardDynamicsContactsDirect (...)`.

Algorithm 5: Code example for the creation of a constraint set that adds an external point contact to body with id `body_3_id`. The point is located at $(0, 0, -1)^T$ in the local coordinates of the body.

```
ConstraintSet CS;

CS.AddConstraint (body_3_id,
  Vector3d (0., 0., -1.),
  Vector3d (1., 0., 0.)
);
CS.AddConstraint (body_3_id,
  Vector3d (0., 0., -1.),
  Vector3d (0., 1., 0.)
);
CS.AddConstraint (body_3_id,
  Vector3d (0., 0., -1.),
  Vector3d (0., 0., 1.)
);

CS.Bind(model);

ForwardDynamicsContactsDirect (model, q, qdot, tau, CS, qddot);

std::cout « qddot.transpose() « std::endl;
std::cout « CS.force.transpose() « std::endl;
```

3.5.4 Structure Exploiting Spatial Algebra

The pseudo code presented in Algorithm 1 and Algorithm 2 operate on 6-D spaces, but faster formulations can be derived for many spatial operations Featherstone (2008).

An expression like $\boldsymbol{X}\boldsymbol{v}$ would cost 36 multiplications and 30 additions using 6-D arithmacy. However the same operation can be performed by only 24 multiplications and 18 additions which can be seen by rewriting the multiplication using:

$$\begin{bmatrix} \boldsymbol{E} & \boldsymbol{0} \\ -\boldsymbol{E}\boldsymbol{r}\times & \boldsymbol{E} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_O \end{bmatrix} = \begin{bmatrix} \boldsymbol{E}\boldsymbol{\omega} \\ -\boldsymbol{E}\boldsymbol{r}\times\boldsymbol{\omega} + \boldsymbol{E}\boldsymbol{v}_O \end{bmatrix} = \begin{bmatrix} \boldsymbol{E}\boldsymbol{\omega} \\ \boldsymbol{E}(\boldsymbol{v}_O - \boldsymbol{r}\times\boldsymbol{\omega}) \end{bmatrix}. \tag{3.76}$$

The product $\boldsymbol{E}\boldsymbol{\omega}$ costs 9 multiplications and 6 additions, $\boldsymbol{r}\times\boldsymbol{\omega}$ costs 6 multiplications and 3 additions, $\boldsymbol{v}_O - \boldsymbol{v}'$ with $\boldsymbol{v}' = \boldsymbol{r}\times\boldsymbol{\omega}$ costs 3 additions and the multiplication with $\boldsymbol{E}$ costs another 9 multiplications and 6 additions which totals in 24 multiplications and 18 additions.

At various points in the algorithms it is required to concatenate spatial transformations (3.17). Featherstone introduces a compact representation of the spatial transform that only stores the 3×3 orientation matrix $\boldsymbol{E}$ and the 3-D vector $\boldsymbol{r}$ of the linear displacement instead of a full 6×6 matrix. This compact representation is denoted as $\text{plx}(\boldsymbol{E},\boldsymbol{r})$, where "plx" here stands for "Plücker transformation". The product of two spatial transformations

$$\boldsymbol{X}_i\boldsymbol{X}_j = \begin{bmatrix} \boldsymbol{E}_i & \boldsymbol{0} \\ -\boldsymbol{E}_i\boldsymbol{r}_i\times & \boldsymbol{E}_i \end{bmatrix} \begin{bmatrix} \boldsymbol{E}_j & \boldsymbol{0} \\ -\boldsymbol{E}_j\boldsymbol{r}_j\times & \boldsymbol{E}_j \end{bmatrix} \tag{3.77}$$

can then, by exploiting $\boldsymbol{E}_j^T\boldsymbol{r}_i\times\boldsymbol{E}_j = (\boldsymbol{E}_j\boldsymbol{r}_i)\times$, be written as

$$\text{plx}(\boldsymbol{E}_i,\boldsymbol{r}_i)\ \text{plx}(\boldsymbol{E}_j,\boldsymbol{r}_j) = \text{plx}(\boldsymbol{E}_i\boldsymbol{E}_j, \boldsymbol{r}_j + \boldsymbol{E}_j^T\boldsymbol{r}_i), \tag{3.78}$$

which only needs 33 multiplications and 24 additions instead of 216 multiplications and 180 additions. The $\text{plx}(\cdot,\cdot)$ data structure can also be used to efficiently evaluate expressions $\boldsymbol{X}\boldsymbol{v}$, $\boldsymbol{X}^{-1}\boldsymbol{v}$, $\boldsymbol{X}^*\boldsymbol{f}$, and $\boldsymbol{X}^T\boldsymbol{f}$.

Another expression for which drastic improvements can be achieved is

$$^{\lambda_i}\boldsymbol{X}_i^*\boldsymbol{I}_i^c\ {}^i\boldsymbol{X}_{\lambda_i} \tag{3.79}$$

which can be found in the Composite Rigid-Body Algorithm (see Algorithm 2). It transforms the spatial inertia from reference frame of body i to that of body λ_i. When using 6-D operations this results in multiplication of three 6×6 matrices which requires 432 multiplications and 360 additions. An alternative operation can be derived that performs the same computation using 52 multiplications and 56 additions. The faster method requires a compact storage the spatial inertia: the 6×6 matrix

$$\boldsymbol{I}^c = \begin{bmatrix} \boldsymbol{I} & m\boldsymbol{c}\times \\ m\boldsymbol{c}\times^T & m\boldsymbol{1} \end{bmatrix} \tag{3.80}$$

can be stored using 10 floating point values: 6 for the lower triangular part of the symmetric inertia matrix $\boldsymbol{I}$, 3 for $m\boldsymbol{c}\times$, and an additional for m. In Featherstone (2008) this compact structure is called $rbi(m,\boldsymbol{h},lt(\boldsymbol{I}))$ where $lt(\cdot)$ denotes the lower triangular matrix of its argument. Using the $rbi(\cdot,\cdot,\cdot)$ structure furthermore results in fewer operations when rigid body inertias are being summed (10 additions instead of 36).

These structure exploiting operators and compact spatial data structures yield two benefits: fewer instructions and also reduced memory consumption. The latter also results in fewer cache misses on current CPU architectures that further improve performance as fewer values have to be fetched from slower memories.

RBDL implements all of the mentioned compact structures and operators. Additional structures and exploiting operators can be found in Featherstone (2008). It also contains many more practical hints to improve performance.

3.5.5 Reuse of Computed Values

Computing the forward dynamics using (3.1) requires computing $\boldsymbol{H}(\boldsymbol{q})$ and $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$. The former would be computed using the Composite Rigid-Body Algorithm (Algorithm 2) whereas the latter can be obtained using modified Recursive Newton-Euler Algorithm as described in Section

3.4.1. Additional computations are required when computing forward dynamics with external contacts, such as the contact Jacobians and Hessians.

Instead of computing each from scratch one can reuse a considerable amount of values. By first calling the Recursive Newton-Euler Algorithm the values ${}^{i}\boldsymbol{X}_{\lambda_i}$ and $\boldsymbol{S}_i$ are already computed and can be reused by the Composite Rigid-Body Algorithm. Also values for $\boldsymbol{v}_i$ are already computed that are required for the contact Jacobians. Similarly all dependent variables for the contact Hessians are already precomputed.

Both the algorithms, the model structure and the programming interface in RBDL allow the reduction of redundant computations in multiple ways. E.g. in RBDL the function to compute the Jacobian for a point is defined using the following declaration:

```
void CalcPointJacobian (
      Model &model,
      const VectorNd &q,
      unsigned int body_id,
      const Vector3d &point_position,
      MatrixNd &G,
      bool update_kinematics = false
);
```

The last parameter `update_kinematics` has a default value of `false`. If it is not explicitly provided as `true`.

3.5.6 Performance Comparisons with Symbolic Code Generation

To evaluate the efficiency of RBDL we compare it with the symbolic code generation package DYNAMOD (Koch, 2015). The highly advanced code generation package was developed in our research group by Kai Henning Koch and uses similar methods to formulate the rigid multibody dynamics quantities. It therefore is an ideal candidate to compare symbolic code generation with recursive algorithms.

DYNAMOD uses Maple™as computer algebra system and also uses Spatial Algebra and the same algorithms as RBDL to formulate the quantities $\boldsymbol{H}(\boldsymbol{q})$ and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. Both contact Jacobians $\boldsymbol{G}(\boldsymbol{q})$ and contact Hessian $\boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are formulated as 6-D quantities, i.e. they contain both linear and angular parts and they use the origin of the constrained body as the reference point. DYNAMOD generates highly optimized code by aggressively caching intermediate symbolic expressions to ensure that they are only evaluated once for each quantity.

For the performance comparison we used a human model with 34 degrees of freedom which uses the same topology and joint types as the one we use for optimization in the later chapters. We also specified contact constraints on both feet to be able to evaluate performance for systems with external contacts.

We used DYNAMOD to generated code for the model and created a benchmarking program that uses the generated code and RBDL to evaluate the quantities $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, $\boldsymbol{G}(\boldsymbol{q})$, $\gamma(\boldsymbol{q}, \dot{\boldsymbol{q}})$. For the contact quantity evaluation we follow the approach used by DYNAMOD, which evaluates both the contact Jacobian and contact Hessian as 6-D quantities at the origin of the constrained body. RBDL uses the CRBA to evaluate $\boldsymbol{H}(\boldsymbol{q})$ and the modified RNEA to evaluate $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

Additionally to the performance measurement of the individual quantities we also measured the time it takes to evaluate all quantities required either for the unconstrained equation of motion (3.1) and the time it takes to evaluate the quantities for the linear system (3.7) of the constrained dynamics. We refer to the former as "Equation of Motion (EoM) Setup" and the later as "Equation of Motion Contact Setup".

The benchmark is performed by evaluating each of the quantities 1.000.000 times with random values in the range of $[-1, 1] \subset \mathbb{R}$ for the entries of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. As modern CPUs automatically adjust their clock speed depending on the current load and CPU temperature one can get varying results depending on the current condition. To accommodate for this we used three trials with each 1.000.000 evaluations and selected the fastest trial for each computed quantity. We further ensured that both RBDL and the generated code compute the same values by verifying that the relative errors are smaller than 1.0×10^{-15} before we run the benchmarks.

In Figure 3.4 we show the relative performance of RBDL to DYNAMOD's symbolic code generation on different CPUs. At the very top of the figure we list the individual CPUs with the colors we use in the plots below. The computations performed by RBDL are done in two ways: compute all values from scratch, or reuse of computed values. The former are shown in the top plot and the latter in the bottom plot. The computations for "EoM Setup" and "EoM Contact Setup" were only performed in the "reuse" mode and are therefore only displayed in the lower plot.

The results show that reusing values is a crucial aspect of gaining high performance and by doing so RBDL manages to achieve and even exceed the performance of generated code when simulating systems with external contacts. RBDL further may increase its advantage by only evaluating the currently required subset of $\boldsymbol{G}(\boldsymbol{q})$ and $\boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. This could in theory also be done with the generated code. However, this would require large changes in the code generation routines and also require to generate new code.

The clear scope, performance, and ease of use of RBDL makes this library an ideal solution when wanting to model and simulate dynamics for a rigid multibody system. Due to its very few dependencies and permissive license it furthermore can easily be integrated into existing projects.

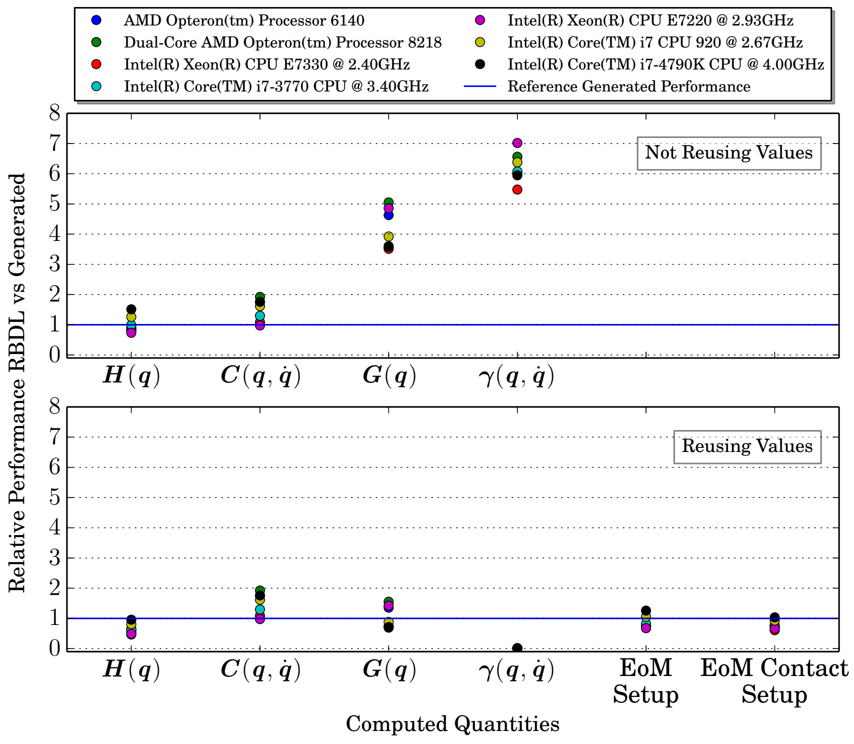


Figure 3.4: Relative Performance of RBDL vs. symbolically generated code on different CPUs without and with reusing of values. Values below 1.0 means that RBDL was faster than the evaluation using generated code. The top plot shows values when RBDL does not reuse any precomputed values, whereas the bottom plot it reuses wherever possible. The values for "EoM Setup" and "EoM Contact Setup" reuse values when computing $\boldsymbol{H}$, $\boldsymbol{G}$, and $\boldsymbol{\gamma}$.

Chapter 4

Subject Specific Modeling of the Dynamics of Human Walking

Modeling the human body using rigid multi-body models is inherently difficult due to the complexity of the human locomotion system. Already the sheer number of joints and bones make simplifications necessary for practical uses. This chapter describes how we create models that we use to analyze human walking movements. One of our aims is to create highly articulated full-body rigid multi-body models that are capable of closely reproducing the recorded motion-capture data from our experiments. We created separate models for the recorded subjects due to significant differences between the subjects.

Creation of these subject-specific rigid multi-body models is very difficult, as crucial information such as locations of joint centers are not available. The recorded motion capture data only consists of spatial movement of markers that we have to relate to segment movement of the human body. Motion capture markers are attached to the subject at specific anatomical landmarks. However, the coordinates of the motion capture markers relative to the body part to which they are attached are unknown and vary from subject to subject. Furthermore the motion capture data in itself is affected by noise due to the non-rigid attachment of the markers to the skin.

Our approach of subject specific modeling consists of two components: HeiMan a highly customizable human meta model based on inertial parameters from biomechanical literature and Puppeteer a graphical motion capture and model editing tool that facilitates creation of virtual markers on the model and fitting of motion capture data onto rigid body models. Both HeiMan and Puppeteer were created in the scope of this thesis.

Furthermore we describe the ground contact model and how the model dynamics are are formulated as ordinary differential equation (ODE).

4.1 Prerequisites and Definitions

Before we can discuss the problem formulation and methods to create subject specific models we introduce some definitions and descriptions for relevant quantities.

Definition 1 (Motion Capture Data, Motion Capture Frame)**.** *Let n_F be the number of motion capture frames, i.e. time instances for which the 3-D locations of all markers were recorded, and let n_M be the number of recorded markers. Then the* motion capture data *is defined as:*

$$\boldsymbol{m}_{ij}^D \in \mathbb{R}^3, i = 1, \ldots, n_F, j = 1, \ldots n_M. \tag{4.1}$$

A motion capture frame *consists of all marker locations for a specified frame index $i, 1 \leq i \leq n_F$.*

Rigid Multi-Body Model, Joints Formulation of rigid multi-body models were described in detail in Section 3.3. Of relevance here is that a rigid multi-body model is a finite set of bodies connected by joints that limit the relative movements of the bodies. Each body defines its own local coordinate system, which is also called the *(local) coordinate system of the body*. Joints and bodies can also be seen as a tree structured graph where the joints are the edges and the bodies are the nodes. Joints are idealized, which means that they only rotate around or translate along predefined axes. For our human models we only have three translational joints for the base body of our model to locate it in a 3D space and all others joints are either 3-DOF spherical or 1-D hinge joints.

In this chapter the terms "rigid body model" or just "model" will always refer to a "rigid multibody model".

Posture, Generalized Coordinates The posture of the rigid multi-body model is described using generalized coordinates $\boldsymbol{q} \in \mathbb{R}^{n_{dof}}$ where $n_{dof} \in \mathbb{N}$ are the number of degrees of freedom. Generalized coordinates contain translations and rotations of all joints variables. $\boldsymbol{q}$ is also referred to as the generalized positions.

Segments When dealing with rigid multi-body models of humans the bodies are also referred to as "segments", i.e. the "pelvis segment" will refer to the rigid body in the model that corresponds to the pelvis of the human.

Definition 2 (Virtual Marker)**.** *A* virtual marker *is a designated point on a body in a rigid multi-body model. It is defined as a pair $\boldsymbol{p}^m = (i, {}^i\boldsymbol{p})$, where i refers to the index of the reference body and ${}^i\boldsymbol{p} \in \mathbb{R}^3$ are the coordinates of the point in the coordinate system of body i.*

The position of the virtual marker $\boldsymbol{p}^m_j = (i, {}^i\boldsymbol{p}_j)$ in the global reference frame is defined as:

$$ {}^0\boldsymbol{p}_j = \boldsymbol{m}^M_j(\boldsymbol{q}, \boldsymbol{p}^m_j) := {}^0\boldsymbol{r}_i(\boldsymbol{q}) + {}^0\boldsymbol{E}_i(\boldsymbol{q})\ {}^i\boldsymbol{p}_j \tag{4.2} $$

where $\boldsymbol{q}$ are the generalized positions of the rigid multi-body model, ${}^0\boldsymbol{r}_i(\boldsymbol{q}) \in \mathbb{R}^3$ the location of the origin of body i in global coordinates and ${}^0\boldsymbol{E}_i(\boldsymbol{q}) \in \mathbb{R}^{3\times 3}$ the orientation of body i relative to the global coordinate frame.

Virtual markers are used to define a mapping from the movement of the motion capture markers to the corresponding movement of virtual markers.

4.2 Problem Description

For a given model topology, i.e. segments and degrees of freedom for the individual joints, we would like to determine joint centers $\boldsymbol{p}^c \in \mathbb{R}^3$ and locations of virtual markers $\boldsymbol{p}^m$ such that the model can closely reproduce the recorded motion capture data $\boldsymbol{m}^D_{ij}$.

The values of $\boldsymbol{p}^c$ and $\boldsymbol{p}^m$ have to be obtained for each subject individually due to differences between the subjects, which we will show in the following section.

4.2.1 Subject Specific Differences

Differences in Subject Dimensions

In Figure 4.1 we show a single motion capture frame for all subjects from the motion capture experiments. The displayed capture frames are used from the "Standing" calibration motions and all subjects are displayed in the neutral standing pose. The snapshots all use the same camera parameters (location, orientation and zoom factor) and the camera used orthographic projection, which preserves distances between markers along the horizontal and longitudinal axis.

As can be seen from Figure 4.1 the subjects vary visibly in height but also less visible in width. Using a single rigid multi-body model for all subjects would therefore be only a coarse approximation of the model.

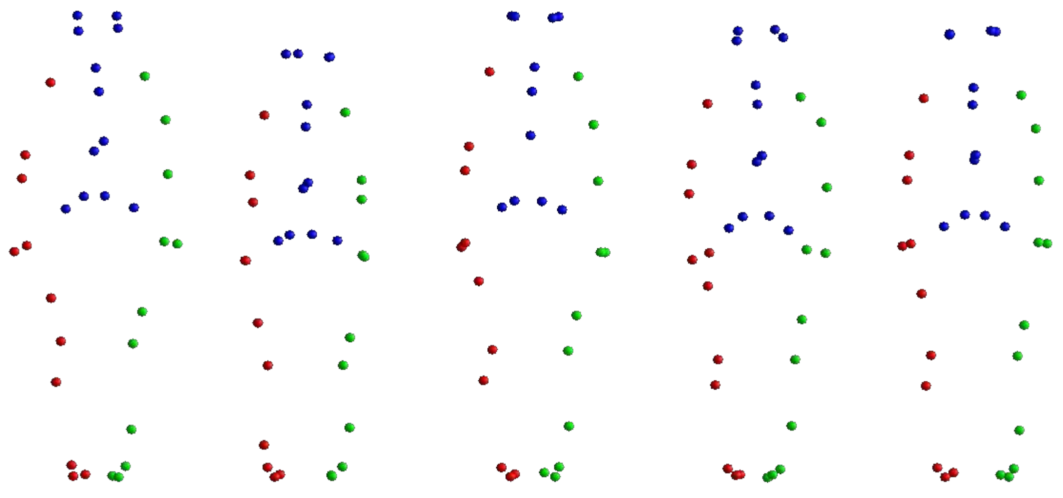

(a) Subject1 (b) Subject2 (c) Subject3 (d) Subject4 (e) Subject5

Figure 4.1: Motion capture data of all subjects while standing in neutral pose shown in frontal view. The displayed images were rendered in a way such that distances along the horizontal and longitudinal axes are the same for all images.

Differences in Segment Lengths and Marker Movements

Instead of using a single model for all subjects one could also create models using data from biomechanical literature such as de Leva (1996) or Dumas et al. (2007). They present tables that define segment dimensions, joint center locations, segment masses, and inertial parameters that are influenced by gender, height, and weight of the subject. However joint locations are only influenced by the total height of the subject as the all segment lengths and therefore joint locations are scaled uniformly.

However there are differences in segment lengths that are not captured by this uniform scaling approach and differ from subject to subject. In Figure 4.2 we have highlighted the markers *RSHO* and *RELB*, which are placed near anatomical landmarks of the right shoulder and elbow joints. Similarly, we have highlighted the markers *RKNE* and *RANK*, which are close to the right knee and ankle joints. These two marker pairs are good approximations for the segment lengths of the upper arm (*RSHO-RELB*) and shank (*RKNE-RANK*) as the designated marker locations are well defined and easy to locate.

Table 4.1 shows average distance and standard deviation of the right shoulder and right elbow markers *RSHO* and *RELB* along with the relative average distance normalized by subject height. Similarly Table 4.2 shows these values for the right knee and right ankle markers *RKNE* and *RANK*. The average distances and standard deviations were obtained from a neutral motion capture trial for each subject.

When looking at the relative average distances one can see a tendency for a shared factor for each distance. Nevertheless for each segment there are outliers, e.g. Subject5 for the *RSHO-RELB* distance and Subject1 for the *RKNE-RANK* distance. This suggest that uniform scaling of segments based on subject height are only of limited use for precise models.

To summarize, creating rigid multi-body models from motion capture marker data is inherently difficult due to multiple reasons. Some of them are:

1. **Non-rigid movement of the recorded markers** due to movement of the skin caused by skin deformation caused by muscle contraction or skin movement in general versus rigid body movement in the rigid multi-body body model.

2. **Unknown locations of joint centers** (and related: unknown segment dimensions),

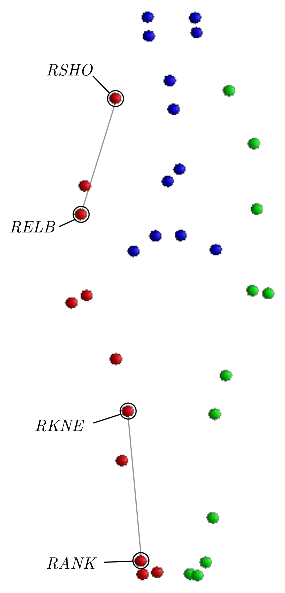


Figure 4.2: Markers used for distance computations listed in Table 4.1 and Table 4.2.

which are required for a rigid body model.

3. **Inherently highly complex joints** in the human body (especially spine and shoulder) and idealized and fewer joints in the rigid body model.

	Average Distance *RSHO - RELB*	Subject Height	Distance / Height
Subject1	$384.10 \pm 0.91mm$	$1820mm$	0.2110 ± 0.0005
Subject2	$335.80 \pm 2.51mm$	$1690mm$	0.1987 ± 0.0015
Subject3	$391.91 \pm 1.18mm$	$1860mm$	0.2107 ± 0.0006
Subject4	$344.51 \pm 3.66mm$	$1790mm$	0.1925 ± 0.0020
Subject5	$315.60 \pm 0.44mm$	$1800mm$	0.1753 ± 0.0002

Table 4.1: Mean and standard deviation of the marker distances *RSHO* (right shoulder) and *RELB* (right elbow), subject heights, and average marker distance normalized by subject height for each subject.

4.2.2 Subject Specific Modeling as a Parameter Estimation Problem

Formally, an optimal subject specific model can be described by the following optimization problem:

Definition 3 (Optimal Subject Specific Model)**.** *Let n_F be the number of recorded motion capture frames, n_M be the number of markers used, n_J be the number of joints in for which the locations are unknown. Let further $\boldsymbol{p}_k^m \in \mathbb{R}^3, k = 1, \dots, n_M$ be the virtual markers of the model, $\boldsymbol{p}_l^c \in \mathbb{R}^3, l = 1, \dots, n_J$ the joint center locations of the unknown joint centers described in local coordinates, $\boldsymbol{q}_i \in \mathbb{R}^{n_{dof}}, i = 1, \dots, n_F$ be the generalized positions for the model at frame i. Let $\boldsymbol{m}_{i,j}^D \in \mathbb{R}^3$ be the marker position of marker j at capture frame i and $\boldsymbol{m}_j^M(\boldsymbol{q}_i, \boldsymbol{p}^c, \boldsymbol{p}^m) \in \mathbb{R}^3$ be the position of the model marker j for the model posture $\boldsymbol{q}_i \in \mathbb{R}^{n_{dof}}$ at capture frame i. Let $\boldsymbol{p}^m$ denote all local model markers and $\boldsymbol{p}^c$ all joint center coordinates.*

	Average Distance *RKNE - RANK*	Subject Height	Distance / Height
Subject1	$475.84mm \pm 6.21mm$	$1820mm$	0.2615 ± 0.0034
Subject2	$401.53mm \pm 3.69mm$	$1690mm$	0.2376 ± 0.0022
Subject3	$461.92mm \pm 2.98mm$	$1860mm$	0.2483 ± 0.0016
Subject4	$419.57mm \pm 3.76mm$	$1790mm$	0.2344 ± 0.0021
Subject5	$433.06mm \pm 5.14mm$	$1800mm$	0.2406 ± 0.0029

Table 4.2: Mean and standard deviation of the marker distances *RKNE* (right knee) and *RANK* (right ankle), subject heights, and average marker distance normalized by subject height for each subject.

The optimal subject specific rigid body model is then described by the optimal solution of

$$\min_{\boldsymbol{q}_1,\dots,\boldsymbol{q}_{n_F},\boldsymbol{p}^m,\boldsymbol{p}^c} \sum_{i=1}^{n_F}\sum_{j=1}^{n_M} \frac{1}{2}||\boldsymbol{m}^D_{i,j} - \boldsymbol{m}^M_j(\boldsymbol{q}_i, \boldsymbol{p}^m, \boldsymbol{p}^c)||^2_2. \tag{4.3}$$

Solving the parameter estimation problem (4.3) is difficult as the parameters $\boldsymbol{q}_i, \boldsymbol{p}^c$, and $\boldsymbol{p}^m$ are redundant. For example, a nonzero displacement of a joint center can be negated by a displacement of the local marker coordinates in the affected segment and an appropriate change of joint variables.

The solution of (4.3) may also not be uniquely defined. For joints that rotate around a single axis such as elbow or knee joints the possible joint centers are all on the axis of rotation.

If some of the variables are known the is problem reduced to problems that are easier to solve. E.g. if $\boldsymbol{p}^c$ and $\boldsymbol{p}^m$ are given, the problem is reduced to an inverse kinematics problem. Similarly, if $\boldsymbol{q}_i$ and $\boldsymbol{p}^c$ are given, the local marker coordinates can be deduced as the mean coordinates of the markers from data transformed into the local space of the segments.

Some of the issues can be alleviated by adding regularization terms, however these may add unwanted biases to the solution. E.g. adding minimization of the local coordinates $\boldsymbol{p}^m$, the origins of the segments will tend to be in the centroid of the markers. Therefore great care has to be taken and manual scaling of the additional terms is required.

Furthermore the problem (4.3) quickly becomes very large. The number of free optimization variables n dependent on the number of frames n_F, number of unknown joint locations n_J, and number of local marker coordinates n_M is:

$$n = n_F n_{dof} + 3n_J + 3n_M \tag{4.4}$$

Directly solving this problem is not advisable, since the high number of degrees of freedom for the model $n_{dof} = 35$ prevents use of the high numbers of motion capture frames n_F that would be necessary to get a good fit.

4.2.3 Related Problems and Methods

Both in biomechanics and computer graphics literature related problems are discussed. There is no clear classification of the individual problems, nevertheless we want to name a few: *Joint center estimation* tries to find the center of rotation for two connected bodies from two sets of points (Umeyama, 1991), *joint parameter estimation* computes joint locations and the axes of movement from motion capture data (Schwartz and Rozumalski, 2005). The closest related problem to ours is that of *skeleton estimation* or sometimes called *skeleton fitting*. It tries to identify individual rigid body movements and a skeleton hierarchy from motion capture data.

In clinical gait analysis an often used method is the Plug-in Gait (PiG) model that is used by VICON$^{\text{TM}}$ motion capture system. It uses a specific set of motion capture markers (a so-called *marker set*) that are placed on specific points on the body. Subsets of the markers are then used to define reference frames for individual segments that are computed for each recorded motion capture frame. A limitation of this approach is that the individual segments are not connected with proper spherical or hinge joints, instead they also allow relative translations. The lower-limb kinematics of PiG is based on Davis et al. (1991), however we were not able to find any references that describe how the PiG upper-limb kinematics are computed.

The skeleton estimation method in O'Brien et al. (2000) computes the joint parameters of a hierarchical system automatically using magnetic motion capture data. For two connected segments it computes the shared point or axes (in case of a 1-DOF joint) of both bodies that moves the least. It takes advantage of the magnetic motion capture data which provides not only position but also orientation of the sensor. For best results the recorded human should exercise all its degrees of freedom.

Kirk et al. (2005) describe a method that uses a conventional marker-based motion capture system. It first clusters the data into groups of points that undergo similar movements and is then able to automatically compute a model topology using a spanning tree. The edges of this tree are the joints and the nodes are the marker groups. It then computes a rigid body model by traversing the spanning tree from the root to the leafs to compute joint centers. For two marker groups A and B they use the method proposed in Horn (1987) to compute the joint center that best describes the movement of the markers in B relative to the coordinate system defined for A. This is done for every motion capture frame and the average joint center of all frames is then used to define the joint center of the rigid body model. The method computes both joint centers and marker coordinates in the local space of the segments and the motions can be mapped onto the model using inverse kinematics.

4.2.4 Our Approach

The method described in Kirk et al. (2005) is very attractive as it works with standard motion capture marker data. However, the method assumes that the connecting joints have full spherical movement. Joints with single degrees of freedom such as knees or elbows are not supported by this method and would require further adjustments or the mapped motion would have to be re-targeted onto a model with the actual degrees of freedom.

In our approach we solve the problem by the combination of two separate parts: HeiMan, a highly parameterized and customizable rigid body model, and Puppeteer, a graphical model editor and motion capture fitting tool. Starting with a default HeiMan model we adjust the model and create virtual markers using Puppeteer, for which we implemented a robust inverse kinematics method to compute the joint angles. Puppeteer directly operates on the reduced coordinates of the model which alleviates us from any post processing or additional motion re-targeting. Both HeiMan and Puppeteer were created for this thesis.

4.3 The HeiMan Model - a Highly Customizable Rigid Body Model for Humans

HeiMan is a highly parameterized rigid multi-body model for human models. It allows creating a wide variety of models by use of tabular data reported in de Leva (1996) and Dumas et al. (2007). A schematic overview is given in Figure 4.3, which shows the available segments, joints, and also the overall model topology that spans from the pelvis.

The HeiMan model is based on the biomechanical data by de Leva (de Leva, 1996) for joint center locations, segment masses and segment inertias. The data can be used to create models by specifying gender, model height, and model weight by scaling the individual joint center locations, masses, and inertias. The data assumes that all joints and center of mass locations are on the frontal plane (which is in our case the $Y - Z$ plane). For the lateral displacements of the hip and shoulder joints we used the data reported by Dumas et al. (2007) since the de Leva data does not report these.

HeiMan not only allows creating a wide range of human rigid multi-body models that can be used for dynamics simulation, it also automatically creates specifications for the visualization of the models. A screenshot of a visualization of a HeiMan model using MeshUp is presented in Figure 4.4.

4.3.1 Model Creation

To describe a model using HEIMAN one first has to specify the model height and weight. As of this writing only male models are supported, however HEIMAN can easily be extended by adding biomechanical data for female models.

HEIMAN initializes the default parameters listed in Table 4.3, Table 4.4, and uses the specified model height and weight. A parameter vector $\boldsymbol{p}$, which contains all these parameters is returned. This vector can then be used to create a model, which yields a model that closely conforms to the description of de Leva.

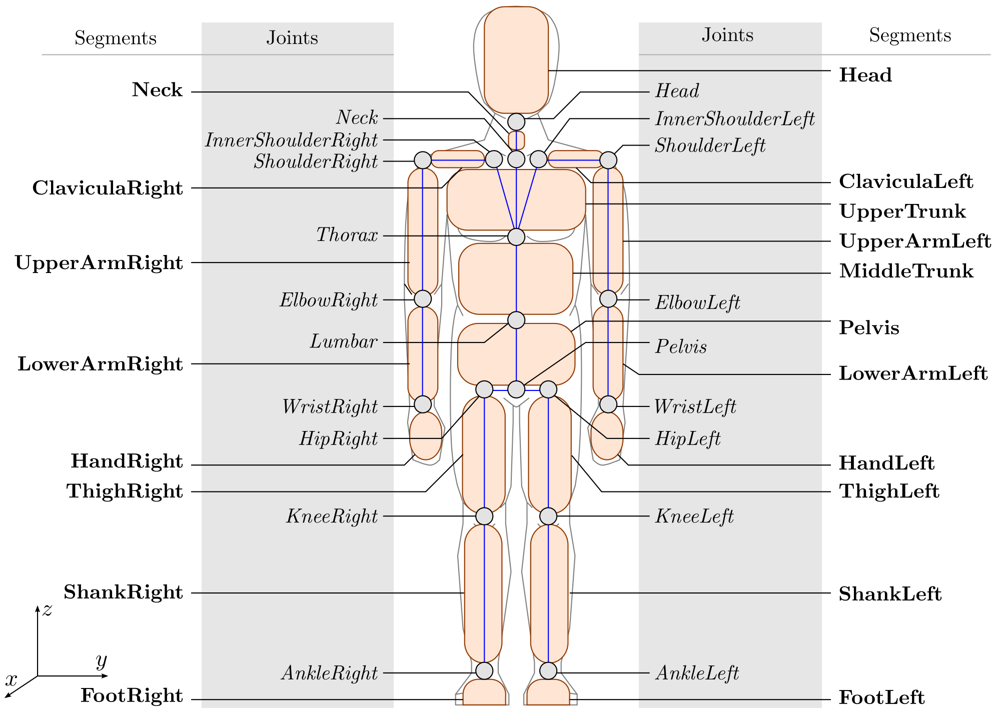


Figure 4.3: Schematic overview of HEIMAN. Joint names are shown in italic font whereas segment names are printed bold. Blue lines show the joint center relative to the parent joint. Segments are shown in light brown.

To create a custom model one can override individual parameters in $\boldsymbol{p}$. E.g. if the vector $\boldsymbol{p}$ is created for a model of height $1.79m$ and mass $76kg$ the default parameter for the pelvis height is $\boldsymbol{p}_{PelvisHeight} = 0.3752 \cdot ModelHeight = 0.1498m$. By setting $\boldsymbol{p}_{PelvisHeight} = 0.3m$ one would get a model where the pelvis segment is $0.3m$ high. A visualization of the default model and the customized model is shown in Figure 4.4.

4.3.2 Joints

The mobility of the model is defined by the degrees of freedom for each joint. By default HEIMAN uses the degrees of freedom described below, however for each joint the actual degrees of freedom can be specified individually. This allows us to create 3-D, 2-D, and even 1-D human models.

In the following we describe all joints that are available in HEIMAN in terms of joint location, degrees of freedom and connectivity to neighbouring segments.

Pelvis The pelvis joint by default has six degrees of freedom: translation along x, y, and z, and rotation around y, x, and z (i.e. YXZ-Cardan angles). The joint is also the origin of the Pelvis segment.

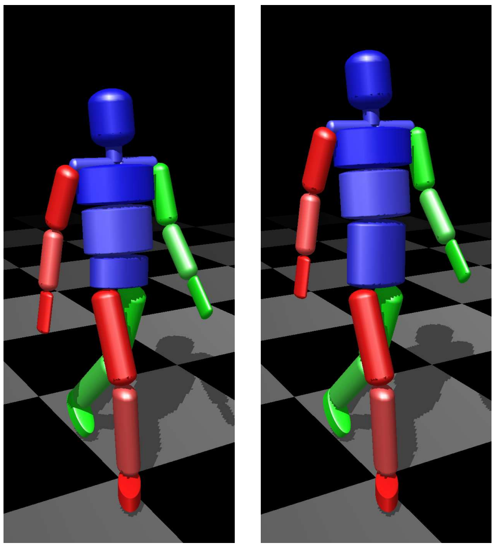

(a) Default model (b) Customized model

Figure 4.4: Visualization of a default HeiMan model for a human of height $1.74m$ and weight of $76kg$ and an adjusted model for which the pelvis height was set to $0.3m$.

HipRight, HipLeft The hip joints are located at $(0, \boldsymbol{p}_{HipRightWidth}, 0)^T$ and $(0, \boldsymbol{p}_{HipLeftWidth}, 0)^T$, in the coordinate frame of the Pelvis segment respectively. By default it has three rotational degrees of freedom around the local y-, x-, and z-axis. The joints are the origins of the segments ThighRight and ThighLeft.

KneeRight, KneeLeft The knee joints are located at $(0, 0, -\boldsymbol{p}_{ThighRightLength})^T$ and $(0, 0, -\boldsymbol{p}_{ThighLeftLength})^T$ in the coordinate frame of ThighRight and ThighLeft, respectively. Each joint has one degree of freedom around the local y-axis and their locations define the origins of the ShankLeft and ShankRight segments.

AnkleRight, AnkleLeft The ankle joints are specified in the local coordinate frame of the ShankRight and ShankLeft segments at the coordinates $(0, 0, -\boldsymbol{p}_{ShankRightLength})^T$ and $(0, 0, -\boldsymbol{p}_{ShankLeftLength})^T$. The ankle joints have the same degrees of freedom as the hip joints, namely the three successive rotations around the y-, x-, and then z-axis. Again, the joints define the origin for the connected segments of FootRight and FootLeft.

Lumbar Joint The Lumbar joint is located in the coordinate frame of the segment Pelvis at the local coordinates $(0, 0, \boldsymbol{p}_{PelvisHeight})^T$. It has three revolute degrees of freedom around the y, x, and z axes. Its location defines the origin of the MiddleTrunk segment

Thorax Joint The Thorax joint is specified in the local coordinate frame of the MiddleTrunk segment at the coordinates $(0, 0, \boldsymbol{p}_{MiddleTrunkHeight})^T$. By default this joint is a fixed joint, which means that there is no relative movement between the UpperTrunk segment and the MiddleTrunk segment. Nevertheless, a separate coordinate frame for the UpperTrunk is still defined at the joint location.

InnerShoulderRight, InnerShoulderLeft The InnerShoulderRight and InnerShoulderLeft joints are defined in the local coordinate frame of the segment UpperTrunk at the coordinates $(0, 0, \boldsymbol{p}_{ShoulderRightHeight})^T$ and $(0, 0, \boldsymbol{p}_{ShoulderLeftHeight})^T$. The joints are fixed by default but define the coordinate frames for the segments ClaviculaRight and ClaviculaLeft.

ShoulderRight, ShoulderLeft The joints ShoulderRight and ShoulderLeft are defined relative to the coordinate frames of ClaviculaRight and ClaviculaLeft at the coordinates $(0, -\boldsymbol{p}_{ClaviculaRightLength}, 0)^T$ and $(0, \boldsymbol{p}_{ClaviculaLeftLength}, 0)^T$. Each has three rotational degrees of freedom around the y-, x-, and then z-axis. The joints define the coordinate frames of the segments UpperArmRight and UpperArmLeft.

ElbowRight, ElbowLeft The elbow joints are located in the coordinate frames of the segments UpperArmRight and UpperArmLeft at the locations $(0, 0, -\boldsymbol{p}_{UpperArmRightLength})^T$ and $(0, 0, -\boldsymbol{p}_{UpperArmLeftLength})^T$, respectively. Each has by default a revolute joint that allows a rotation around the y-axis. The joint locations coincide with the origins of the segments LowerArmRight and LowerArmLeft.

WristRight, WristLeft The wrist joints are specified in the coordinate frames of LowerArmRight and LowerArmLeft at the coordinates $(0, 0, -\boldsymbol{p}_{LowerArmRightLength})^T$ and $(0, 0, -\boldsymbol{p}_{LowerArmLeftLength})^T$. By default the joints are fixed joints.

Neck The Neck joint is defined in the coordinate frame of the segment UpperTrunk at $(0, 0, \boldsymbol{p}_{SuprasternaleHeight})^T$, which coarsely corresponds to the point between the left and right clavicles. It is a fixed joint that does not allow relative movement of the Neck segment relative to the UpperTrunk segment.

Head The Head joint is located in the coordinate frame of the Neck at $(0, 0, \boldsymbol{p}_{NeckLength})^T$, which is at the height of the lower end of the mandible. It has three degrees of freedom around the y-, x-, and z- axis.

4.3.3 Segments

The segments define the dynamic properties of the model while joints define the mobility of the model. Segments are the bodies of the rigid multi-body model and their properties are defined by mass, location of the center of mass and the inertia matrix of the body.

The HeiMan parameters that are used to describe the masses of each segment are listed in Table 4.4. Together with the lengths of the segments listed in Table 4.3 and the relative center of masses and relative radii of gyration listed in Table 4.5 we can compute the absolute center of masses and inertia matrices for each body.

For the UpperTrunk segment the length is given as $l = \boldsymbol{p}_{UpperTrunkHeight} = 0.0980 \cdot \mathit{ModelHeight}$ and its mass as $m = \boldsymbol{p}_{UpperTrunkMass} = 0.1596 \cdot \mathit{ModelMass}$. For a model of height $1.79m$ and mass of $76kg$ we have $l = 0.175m$ and $m = 12.13kg$ and the relative center of mass location is given in Table 4.5 as $\bar{\boldsymbol{r}}^{COM}_{UpperTrunk}(0, 0, 0.7001)^T$. Using this we can compute the absolute center of mass of the UpperTrunk as:

$$\boldsymbol{r}^{COM}_{UpperTrunk} = \bar{\boldsymbol{r}}^{COM}_{UpperTrunk} \cdot l \tag{4.5}$$

$$= \begin{bmatrix} 0 \\ 0 \\ 0.7001 \end{bmatrix} \cdot 0.175m \approx \begin{bmatrix} 0 \\ 0 \\ 0.123m \end{bmatrix} \tag{4.6}$$

in the coordinate frame of the UpperTrunk. Similarly, the center of mass can be expressed and computed for other segments.

The inertia matrix of the segment is computed using the segment mass m, segment length l, and the relative radii of gyration $\bar{\boldsymbol{r}}^{gyr}$ of the segment. The relative radii of gyration of the UpperTrunk listed in Table 4.5 are $\bar{\boldsymbol{r}}^{gyr} = (0.716, 0.454, 0.659)^T$. The inertia matrix of the segment at the center of mass is then given by

$$\boldsymbol{I}^{C}_{UpperTrunk} = \begin{bmatrix} (\bar{\boldsymbol{r}}^{gyr}_x * l)^2 * m & 0 & 0 \\ 0 & (\bar{\boldsymbol{r}}^{gyr}_y * l)^2 * m & 0 \\ 0 & 0 & (\bar{\boldsymbol{r}}^{gyr}_z * l)^2 * m \end{bmatrix}. \tag{4.7}$$

Analogously we can express the inertia matrix for all segments of the HeiMan model.

Parameter Name	**Default Value** (relative to the total height of the subject)	**Source**
ClaviculaLeftLength	0.1200	Dumas et al. (2007)
ClaviculaRightLength	0.1200	Dumas et al. (2007)
FootLeftHeight	0.0468	guess
FootLeftLength	0.1462	guess
FootLeftPosteriorPoint	0.0400	guess
FootRightHeight	0.0468	guess
FootRightLength	0.1462	guess
FootRightPosteriorPoint	0.0400	guess
HandLeftLength	0.1079	de Leva (1996)
HandRightLength	0.1079	de Leva (1996)
HeadLength	0.1168	de Leva (1996)
HipLeftWidth	0.0461	Dumas et al. (2007)
HipRightWidth	0.0461	Dumas et al. (2007)
LowerArmLeftLength	0.1545	de Leva (1996)
LowerArmRightLength	0.1545	de Leva (1996)
PelvisHeight	0.0837	de Leva (1996)
MiddleTrunkHeight	0.3752	de Leva (1996)
NeckLength	0.0638	de Leva (1996)
ShankLeftLength	0.2529	de Leva (1996)
ShankRightLength	0.2529	de Leva (1996)
ShoulderLeftHeight	0.0980	de Leva (1996)
ShoulderRightHeight	0.0980	de Leva (1996)
SuprasternaleHeight	0.0980	de Leva (1996)
ThighLeftLength	0.2425	de Leva (1996)
ThighRightLength	0.2425	de Leva (1996)
UpperArmLeftLength	0.1618	de Leva (1996)
UpperArmRightLength	0.1618	de Leva (1996)
UpperTrunkHeight	0.0980	de Leva (1996)

Table 4.3: Parameter for segment lengths and joint center locations and their default values. By default the parameters are uniformly scaled by the total subject height but can be individually overridden.

4.3.4 Implementation Notes

HeiMan is implemented as a Lua (Ierusalimschy et al., 2006) script. We have created and formalized a textual file format called LuaModel[1] that uses the syntax of the Lua. The LuaModel file format allows description of rigid multi-body models together with additional information such as the coordinate system of the global coordinate frame, visualizations of the bodies and arbitrary custom meta data.

The syntax of Lua is very clear and is easy to read and learned. By using the Lua syntax we also have all possibilities of the scripting language. This includes the definition of functions but most importantly it allows defining variables that facilitate the creation of parameterized

[1]The specification of the LuaModel format can be found at `https://bitbucket.org/MartinFelis/meshup/src/default/doc/FormatDescriptions.md`

Parameter Name	**Default Value** (relative to the total mass of the subject)	**Source**
PelvisMass	0.1117	de Leva (1996)
ThighRightMass	0.1416	de Leva (1996)
ShankRightMass	0.0433	de Leva (1996)
FootRightMass	0.0137	de Leva (1996)
ThighLeftMass	0.1416	de Leva (1996)
ShankLeftMass	0.0433	de Leva (1996)
FootLeftMass	0.0137	de Leva (1996)
MiddleTrunkMass	0.1633	de Leva (1996)
UpperTrunkMass	0.1596	de Leva (1996)
ClaviculaRightMass	–	-
UpperArmRightMass	0.0271	de Leva (1996)
LowerArmRightMass	0.0162	de Leva (1996)
HandRightMass	0.0061	de Leva (1996)
ClaviculaLeftMass	–	-
UpperArmLeftMass	0.0271	de Leva (1996)
LowerArmLeftMass	0.0162	de Leva (1996)
HandLeftMass	0.0061	de Leva (1996)
NeckMass	–	-
HeadMass	0.0694	de Leva (1996)

Table 4.4: Parameters for the individual segment masses of HEIMAN. The masses are specified relative to the model mass but individual values can be overridden manually. For the segments *ClaviculaRight*, *ClaviculaLeft*, and *Neck* no data was available. By default HEIMAN uses a mass of 0 for these segments.

models. E.g. a variable that describes the length of a segment can be reused when defining the dimensions of the visualization of the segment.

Models described using LUAMODEL can be directly loaded by RBDL, the rigid body visualization tool MESHUP[2], and our subject specific modeling tool PUPPETEER, which will be described in the following section.

4.4 Puppeteer - Subject Specific Modeling Tool

To create subject-specific models and virtual markers we created PUPPETEER. PUPPETEER is a motion capture visualization and rigid multi-body modeling tool. It allows visually adjusting the model by simultaneously visualizing rigid body models and motion capture data.

A screenshot of PUPPETEER is shown in Figure 4.5. It shows the 3-D view in the center which displays the model using three dimensional geometric objects and the motion capture data as coloured points in the front. The displayed motion capture frame can be chosen using the slider at the bottom of the tool. Furthermore, the motion capture data can be played back in real-time.

To create a subject-specific model for given motion capture data one first needs to decide which virtual markers are assigned to which segment. This mapping is only about the *association* of a virtual marker to a segment and not about the actual local coordinates of the marker. In this work we use the virtual marker set mapping listed in Table 4.6, but PUPPETEER can be freely adjusted for other marker names and mappings.

For a given virtual marker set mapping and motion capture data one proceeds as follows to

[2]`https://bitbucket.org/MartinFelis/meshup`

Segment Name	**Relative COM**	**Relative Radii of Gyration**		
	Location	**Sagittal**	**Transvers.**	**Longitud.**
Pelvis	$(0, 0, 0.3884)^T$	1.615	0.551	0.587
Thigh	$(0, 0, -0.4095)^T$	0.329	0.329	0.149
Shank	$(0, 0, -0.4458)^T$	0.255	0.249	0.103
Foot	$(0, 0, -0.4415)^T$	0.257	0.245	0.124
MiddleTrunk	$(0, 0, 0.5498)^T$	0.482	0.383	0.468
UpperTrunk	$(0, 0, 0.7001)^T$	0.716	0.454	0.659
Clavicula	–	–	–	–
UpperArm	$(0, 0, -0.5772)^T$	0.285	0.269	0.158
LowerArm	$(0, 0, -0.4574)^T$	0.276	0.265	0.121
Hand	$(0, 0, -0.3624)^T$	0.288	0.235	0.184
Neck	–	–	–	–
Head	$(0, 0, 0.4024)^T$	0.362	0.376	0.312

Table 4.5: Relative Center of Mass locations for each segments and the relative radii of gyration for all segments of HeiMan. All values are taken from de Leva (1996). The relative Center of Mass locations multiplied with the segment length (see Table 4.3) give the locations in local coordinates. The relative radii of gyration are scaling factors that, together with the segment length and segment mass, are used to compute the total segment inertia. No parameters are available for the segments *Clavicula* and *Neck*. By default these segments have their Center of Mass at $(0, 0, 0)^T \in \mathbb{R}^3$ and an inertia of $\mathbf{0} \in \mathbb{R}^{3\times3}$.

create a subject-specific model.

1. Create a basic model using HeiMan by specifying the measured total height and total mass of the subject.

2. Load the model and a motion capture data sequence into Puppeteer.

3. Create virtual markers for the model:

   (a) Choose a motion capture frame i that allows the estimation of the posture of a segment for the subject.

   (b) Adjust $\boldsymbol{q}$ to pose the model so that it mimics the posture of the subject in the chosen motion capture frame.

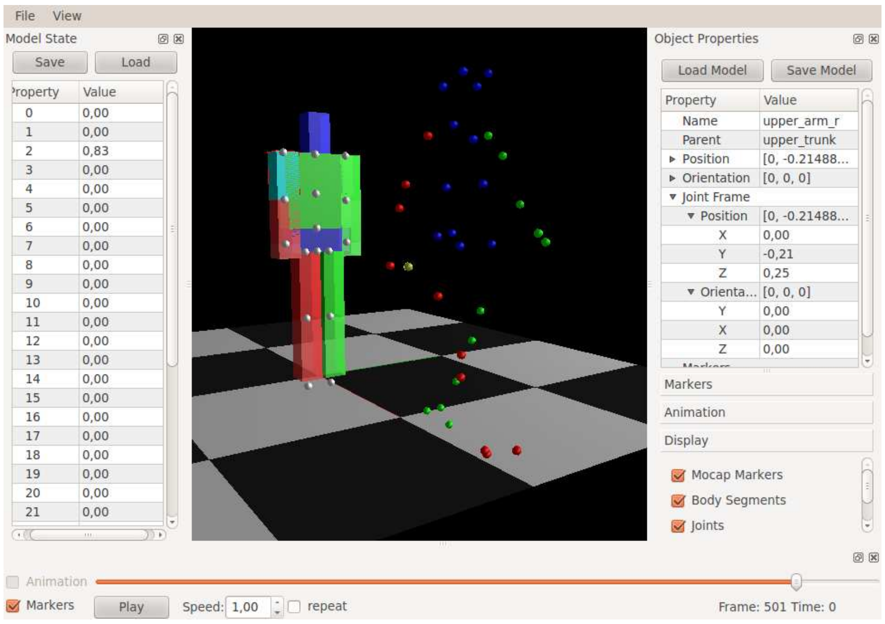
   (c) Adjust joint center locations $\boldsymbol{p}^c$ of model so that joint center locations of the model approximate the estimated joint centers of the subject.

   (d) Assign markers $\boldsymbol{m}_{ij}^D$ from the motion capture frame to the model to create the virtual markers $\boldsymbol{p}_j^m$.

4. Extract length parameters from the adjusted model to create a subject specific HeiMan model.

It can be difficult to find a single motion capture frame that allows for a good marker assignment of all segments. Therefore step 3 usually needs to be performed repeatedly for different motion capture frames.

E.g. to obtain a good estimate for a knee joint center it is easier to locate it for a motion capture frame where the knee flexion of the subject can be estimated. Usually there is no single motion capture frame for both left and right knee which is why this step is better done using multiple motion capture frames.



Property	Value
0	0,00
1	0,00
2	0,83
3	0,00
4	0,00
5	0,00
6	0,00
7	0,00
8	0,00
9	0,00
10	0,00
11	0,00
12	0,00
13	0,00
14	0,00
15	0,00
16	0,00
17	0,00
18	0,00
19	0,00
20	0,00
21	0,00

Property	Value
Name	upper_arm_r
Parent	upper_trunk
▸ Position	[0, -0.21488...
▸ Orientation	[0, 0, 0]
▾ Joint Frame	
▾ Position	[0, -0.21488...
X	0,00
Y	-0,21
Z	0,25
▾ Orienta...	[0, 0, 0]
Y	0,00
X	0,00
Z	0,00

[✓] Mocap Markers
[✓] Body Segments
[✓] Joints

[ ] Animation
[✓] Markers
Speed: 1,00 [ ] repeat
Frame: 501 Time: 0

Figure 4.5: Screenshot of PUPPETEER. We created this tool specifically to adjust rigid multi-body models to approximate the kinematic structure of the recorded subjects. The model is shown using coloured polygons whereas the motion capture data is shown using small coloured points. On the left the values of the generalized positions of the model are shown, on the right the properties of the selected segment (in this image: UpperArmRight). The slider on the bottom is used to select the current motion capture frame.

4.4.1 Adjustment of the Model

Modifying the both posture $\boldsymbol{q}$ and joint center locations $\boldsymbol{p}^c$ in a way that the model joint centers align with joint centers estimated from marker data is what we refer to as "model adjustment". The model adjustment is performed by first adjusting the generalized positions $\boldsymbol{q}$ of the model such that its pose reflects that of the subject in the motion capture data of the selected frame and then adjusting the locations of the joints $\boldsymbol{p}^c$. Figure 4.6 shows a screenshot of PUPPETEER while editing the upper right arm.

Posing the Model

The posture of the model can be adjusted using the model state editor which is highlighted using a red box in Figure 4.6. It allows the direct adjustment of each degree of freedom of the model. Any change is directly applied to the model in the 3-D view, which allows for a fast assessment of the similarity of the model pose and the pose of the subject at the current motion capture frame.

To pose the model one should start using the generalized positions of the pelvis segment and then proceed outwards along the spanning tree of the model.

Modification of Joint Frames

In PUPPETEER the model is adjusted by altering the coordinates of the joint center of a child body. E.g. the pelvis has three child bodies: the right thigh, the left thigh, and the middle trunk. Each of these child bodies is connected with a joint to the pelvis and the location of this joint is specified in coordinates in the pelvis frame. But also the default orientation of the joint can be adjusted. Location and orientation of the joint therefore define a frame which we refer to as the *joint frame*.

E.g. to modify the joint frame of the right shoulder joint one first selects the child body (in this case the right upper arm). This will load all properties of the right upper arm into the

Segment	**Virtual Markers**
Pelvis	*LPSI, LASI, RPSI, RASI*
ThighRight	*RTHI*
ShankRight	*RKNE, RTIB*
FootRight	*RANK, RHEE, RTOE*
ThighLeft	*LTHI*
ShankLeft	*LKNE, LTIB*
FootLeft	*LANK, LHEE, LTOE*
MiddleTrunk	*T10, STRN*
UpperTrunk	*CLAV, C7*
UpperArmRight	*RUPA*
LowerArmRight	*RELB, RWRA, RWRB*
UpperArmLeft	*LUPA*
LowerArmLeft	*LELB, LWRA, LWRB*
Head	*RFHD, RBHD, LFHD, LBHD*

Table 4.6: Virtual marker set mapping used in this work.

object editor. This also loads the location and orientation of the joint into the object editor as the connecting joint is associated with the selected segment.

Modification of joint location and orientation is directly applied to the model in the 3-D view so that an assessment of the new position is quickly possible.

4.4.2 Virtual Marker Assignment

Once the model is properly posed and the joint centers are adjusted the markers can be assigned. This is done by selecting the segment and the markers that should be assigned to the segment. By clicking on the "Assign Markers" button shown on the bottom right in Figure 4.7 one adds virtual markers $\boldsymbol{p}^m$ for the selected motion capture markers.

The virtual markers are created by transforming the global coordinates of the currently selected motion capture markers into the local coordinate frame of the selected segment. If segment l and the markers $\boldsymbol{m}_j^D \in \mathbb{R}^3, j = 1, \ldots, n_s$, are selected, the local coordinates of the virtual markers are obtained using

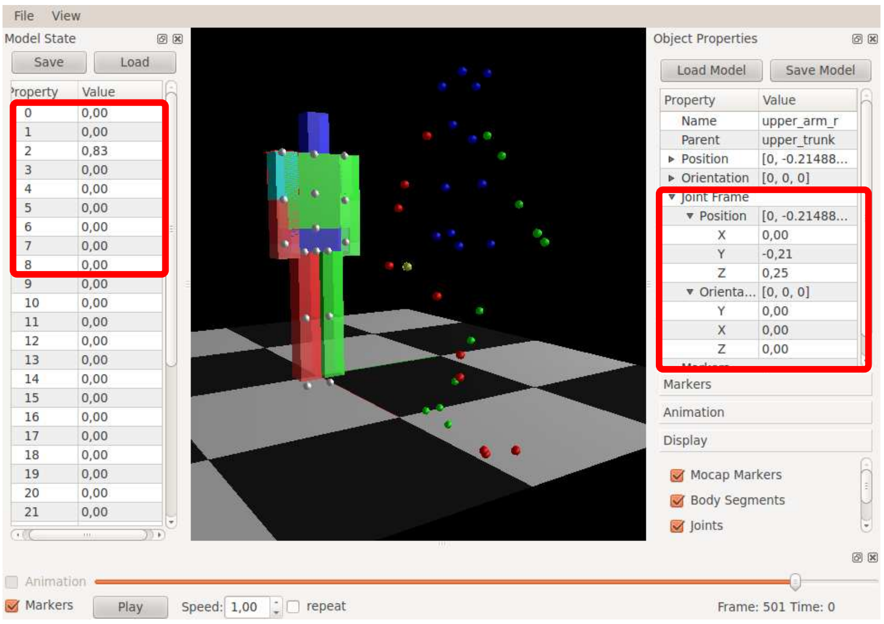
$$^{l}\boldsymbol{p}_j = {}^{l}\boldsymbol{E}_0(\boldsymbol{m}_j^D - {}^{0}\boldsymbol{r}_l), \tag{4.8}$$

where ${}^{0}\boldsymbol{r}_l \in \mathbb{R}^3$ is the location of segment l in the global coordinate frame and ${}^{l}\boldsymbol{E}_0 \in \mathbb{R}^{3\times3}$ the rotation matrix that rotates a coordinate vector from the global coordinate frame to the local coordinate frame of segment l. This then yields the virtual markers $\boldsymbol{p}_j^m = (l, {}^{l}\boldsymbol{p}_j), j = 1, \ldots, n_s$.

4.4.3 Fitting Motion Capture Data to the Subject Model

Once a model is adjusted and virtual markers have been created, the values of $\boldsymbol{p}^c$ and $\boldsymbol{p}^m$ are fully defined and (4.3) is reduced to an inverse kinematics problem. It allows fitting of motion capture data to the model by using a least-squares optimization on the difference of the virtual markers and the motion capture markers. For this we use the method presented in Sugihara (2009), which we briefly describe here.

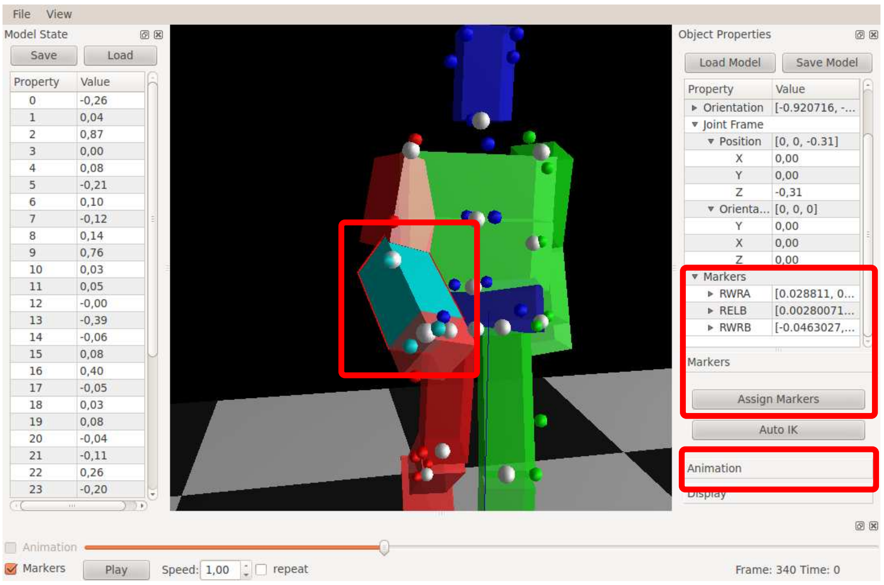
Inverse kinematics computes joint angles $\boldsymbol{q}$ for an articulated model such that designated points of the model coincide or are close to specific points in space. In our case the model is the subject-specific model, the designated points are the virtual markers that should be close to the positions of the motion capture markers for a given motion capture frame.



Property	Value
0	0,00
1	0,00
2	0,83
3	0,00
4	0,00
5	0,00
6	0,00
7	0,00
8	0,00
9	0,00
10	0,00
11	0,00
12	0,00
13	0,00
14	0,00
15	0,00
16	0,00
17	0,00
18	0,00
19	0,00
20	0,00
21	0,00

Property	Value
Name	upper_arm_r
Parent	upper_trunk
▸ Position	[0, -0.21488...
▸ Orientation	[0, 0, 0]
▾ Joint Frame	
▾ Position	[0, -0.21488...
X	0,00
Y	-0,21
Z	0,25
▾ Orienta...	[0, 0, 0]
Y	0,00
X	0,00
Z	0,00

Figure 4.6: Adjustment of posture (left highlighted area) and joint center locations (right highlighted area). The posture of the model can be modified using the model state editor at the left side of the image. Joint center locations can be modified using the object property editor on the right side.



Property	Value
0	-0,26
1	0,04
2	0,87
3	0,00
4	0,08
5	-0,21
6	0,10
7	-0,12
8	0,14
9	0,76
10	0,03
11	0,05
12	-0,00
13	-0,39
14	-0,06
15	0,08
16	0,40
17	-0,05
18	0,03
19	0,08
20	-0,04
21	-0,11
22	0,26
23	-0,20

Property	Value
▸ Orientation	[-0.920716, -...
▾ Joint Frame	
▾ Position	[0, 0, -0.31]
X	0,00
Y	0,00
Z	-0,31
▾ Orienta...	[0, 0, 0]
Y	0,00
X	0,00
Z	0,00
▾ Markers	
▸ RWRA	[0.028811, 0...
▸ RELB	[0.00280071...
▸ RWRB	[-0.0463027,...

Figure 4.7: Marker assignment in PUPPETEER. Virtual markers for a model segment are created by selecting a model segment and markers from the motion capture data in the 3-D view in the center. Using the "Assign Markers" button on the right the current marker positions are transformed into local coordinates of the segment and added as virtual markers to the segment.

The inverse kinematics problem for n_M virtual markers can be described as a weighted least-squares problem:

$$\min_{\boldsymbol{q}} R(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{r}(\boldsymbol{q})^T \boldsymbol{W}_m \boldsymbol{r}(\boldsymbol{q}) \tag{4.9}$$

with a diagonal weighting matrix $\boldsymbol{W}_m = diag\{w_{m,j}\}, w_{m,j} > 0, j = 1, \ldots, 3n_M$ and where

$$\boldsymbol{r}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{r}_1(\boldsymbol{q}) \\ \boldsymbol{r}_2(\boldsymbol{q}) \\ \vdots \\ \boldsymbol{r}_{n_M}(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{m}^D_{f,1} - \boldsymbol{m}^M_1(\boldsymbol{q}) \\ \boldsymbol{m}^D_{f,2} - \boldsymbol{m}^M_2(\boldsymbol{q}) \\ \vdots \\ \boldsymbol{m}^D_{f,n_M} - \boldsymbol{m}^M_{n_M}(\boldsymbol{q}) \end{bmatrix} \in \mathbb{R}^{3n_M} \tag{4.10}$$

are the residuals between the motion capture markers and the virtual markers. The weighting matrix $\boldsymbol{W}$ can be used to prioritize certain markers.

The Jacobian of (4.10) is defined by

$$\boldsymbol{J}(\boldsymbol{q}) = \left[\frac{\partial \boldsymbol{r}_i}{\partial \boldsymbol{q}_j}\right] = \begin{bmatrix} \nabla \boldsymbol{r}_1(\boldsymbol{q})^T \\ \nabla \boldsymbol{r}_2(\boldsymbol{q})^T \\ \vdots \\ \nabla \boldsymbol{r}_{n_M}(\boldsymbol{q})^T \end{bmatrix} \tag{4.11}$$

this allows us to define the derivatives of the objective function of the least-squares problem (4.9) as:

$$\nabla R(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{W}_m \boldsymbol{r}(\boldsymbol{q}) \tag{4.12}$$

$$\nabla^2 R(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{W}_m \boldsymbol{J}(\boldsymbol{q}) + \sum_{i=1}^{n_M} \boldsymbol{r}_i(\boldsymbol{q}) \boldsymbol{W}_{m,i} \nabla^2 \boldsymbol{r}_i(\boldsymbol{q}). \tag{4.13}$$

with $\boldsymbol{W}_{m,i} = diag\{w_{m,3(i-1)+1}, w_{m,3(i-1)+2}, w_{m,3(i-1)+3}\}$, which is the submatrix of $\boldsymbol{W}_m$ that contains only the weighting coefficients for the residual of marker i.

A common method to solve nonlinear problems such as (4.10) is to use Newtons method to obtain a stationary point of the problem. By setting $\boldsymbol{J}_k = \boldsymbol{J}(\boldsymbol{q}_k)$ and $\nabla^2 \boldsymbol{R}_k = \nabla^2 R(\boldsymbol{q}_k)$ we can write the corresponding Newton step as:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \left(\nabla^2 \boldsymbol{R}_k\right)^{-1} \boldsymbol{J}_k \boldsymbol{W}_m \boldsymbol{q}_k \tag{4.14}$$

Instead of using the exact value of the Hessian $\nabla^2 \boldsymbol{R}_k$ one can use the Gauss-Newton method, which uses an approximation of the Hessian by omitting the second term in (4.13). This approximation only uses the predefined weighting matrix $\boldsymbol{W}_m$ and the Jacobian $\boldsymbol{J}_k$, which can easily be computed (see Section 3.4.4) and avoids expensive evaluations of second derivatives. The Gauss-Newton approach then yields an iteration of the form:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta \boldsymbol{q}_k, \tag{4.15}$$

with $\Delta \boldsymbol{q}_k$ being the solution of

$$(\boldsymbol{J}_k^T \boldsymbol{W}_m \boldsymbol{J}_k)\Delta \boldsymbol{q}_k = \boldsymbol{J}_k \boldsymbol{W}_m \boldsymbol{q}_k. \tag{4.16}$$

However the linear system (4.16) is only solvable if $\boldsymbol{J}_k$ has full rank. To alleviate this one can use the Levenberg-Marquardt regularization method. Here the step $\Delta \boldsymbol{q}_k$ is obtained by solving the system

$$(\boldsymbol{H}_k + \boldsymbol{W}_d)\Delta \boldsymbol{q}_k = \boldsymbol{J}_k \boldsymbol{W}_m \boldsymbol{q}_k \tag{4.17}$$

with $\boldsymbol{H}_k = \boldsymbol{J}_k^T \boldsymbol{W}_m \boldsymbol{J}_k$ and $\boldsymbol{W}_d = diag\{w_{d,j}\}, w_{d,j} > 0, j = 1, \ldots, 3n_M$ being the regularization term.

Various methods have been proposed for the choice of the regularization term $\boldsymbol{W}_d$, such as Nakamura and Hanafusa (1986); Chan and Lawrence (1988) and others. In Sugihara (2009) multiple methods are compared and further proposes to use:

$$\boldsymbol{W}_d = \boldsymbol{r}(\boldsymbol{q}_k) + \lambda \mathbf{1} \tag{4.18}$$

with $\lambda > 0$, e.g. $\lambda = 1.0 \times 10^{-3}$.

Sugihara applied this method on two benchmark problems, a robot arm with three spherical joints and a humanoid robot with 18 degrees of freedom. The benchmarks included target positions that were unreachable and also target positions where the optimal solution was in singularities of the robots. Overall in both problems his method performed better both in terms of computation time and reduction of the residuals.

4.4.4 Implementation Notes

Puppeteer is implemented using C++ and the cross-platform graphical user interface toolkit Qt[3]. The 3-D rendering is done using the Open Graphics Library (OpenGL) (Shreiner and Group, 2009). Puppeteer extensively uses RBDL, e.g. to transform between global and segment local space and to efficiently compute Jacobian matrices for the inverse kinematics method. It is also used to perform forward kinematics, e.g. to compute the transformations of the visual representations of the segments when the generalized positions are changed.

Models can be loaded from LuaModel files and also written back to LuaModel files. If additional custom values were stored in the original LuaModel file then these values will be preserved when Puppeteer writes the model back to a file. To access and modify values in Lua files we have created the LuaTables++ library.

To fit a complete motion to a model we perform the optimization (4.9) on each motion capture frame individually. For the first motion capture frame the optimization is started with $\boldsymbol{q}_1 = \boldsymbol{0} \in \mathbb{R}^{n_{dof}}$. The subsequent frames are always started with the solution of the previous motion capture frame. The optimization successfully stops when $||\Delta \boldsymbol{q}_k||_2 < 10^{-8}$ or is canceled if no solution was found after 200 steps. In this work we weigh all residuals equally and use an identity matrix for $\boldsymbol{W}_m$ as we do not have any information about the covariances of the markers.

Motion capture data can be loaded using C3D files (Motion Lab Systems, 2008). C3D is a highly versatile public domain binary file format that can be exported from all motion capture systems we are aware of. It allows storing of marker data but also other biomechanical data such as analog EMG (electromyography) data, forces recorded by force platforms, and contact events in one compact file. C3D files also contain information about various parameters such as the recorded frame rate, the recording software etc. Additionally, arbitrary parameters can be stored in a C3D file. Loading of C3D data is done using a custom C3D reader that we integrated in Puppeteer.

Selection of objects is performed using so-called *color picking*. Color picking assigns each object a unique color and renders the scene to a texture. The coordinates of the pointing device are then transformed into coordinates of the texture. The object under the pointing device can then be determined using the color at the texture coordinate of the pointing device.

4.5 Active and Passive Joint Actuation

The HeiMan model can be used in a purely kinematic manner but its main purpose is to use it for dynamics computations and simulation. In the human body the movement of limbs is produced by muscles that are connected via ligaments and tendons to the bones.

We distinguish between active and passive actuation forces. The active actuation forces are the ones that are produced by contraction of muscles or muscle groups. Passive actuation forces are caused by ligaments and tendons that influence the stiffness and also limit the range of motion of the joints.

[3]`http://qt.io`

Modeling the passive parts can be done on many different levels: starting from spring-dampers in the joints to highly complex musculoskeletal models (Nakamura et al., 2005; Matthew Millard and Delp, 2013). Here we use spring-damper elements in the legs to account for the passive forces, since they only have a minor effect on the computational complexity of the model.

For our model the joint forces are specified in terms of generalized forces $\boldsymbol{\tau}(t) \in \mathbb{R}^{n_{dof}}$ that act directly at the joints. However only $n_u = n_{dof} - 6$ of the degrees of freedom are actuated as the position and orientation of the pelvis of the model is indirectly manipulated by the legs due to the interaction of the feet with the ground. This makes our model a *underactuated system.*

We denote $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ as the input controls of our model. Given controls $\boldsymbol{u}(t)$ we can write their resultant generalized forces as

$$\boldsymbol{\tau}_u(t) = \boldsymbol{T}\boldsymbol{u}(t) = \begin{bmatrix} \mathbf{0}_{6\times n_u} \\ \mathbf{1}_{n_u\times n_u} \end{bmatrix} \boldsymbol{u}(t), \tag{4.19}$$

with the $n_{dof} \times n_u$ selection matrix $\boldsymbol{T}$.

The force produced by the passive spring-damper element acting on the degree of freedom with index i can be stated as:

$$\tau_{sd,i}(t) = k_i(q_i^0 - q_i(t)) - b_i(\dot{q}_i(t)) \in \mathbb{R}, \tag{4.20}$$

where $k_i > 0 \in \mathbb{R}$ is the spring constant, q_i^0 is the rest position of the spring, $q_i(t)$ is the position of the joint, $b_i > 0 \in \mathbb{R}$ is the damping constant, and $\dot{q}_i(t)$ is the joint velocity. For joints that do not have a spring-damper we set $\tau_{sd,i}(t) = 0$. By assembling all spring-damper parameters into a single parameter vector $\boldsymbol{p}_{sd} = (k_1, q_1^0, b_1, k_2, q_2^0, b_2, \ldots, k_{n_{actuated}}, q^0_{n_{actuated}}, b_{n_{actuated}})^T$ allows us to write the vector of all passive actuation forces as $\boldsymbol{\tau}_{sd}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{p}_{sd}) \in \mathbb{R}^{n_{dof}}$. The complete actuation can then be written as:

$$\boldsymbol{\tau}(t) = \boldsymbol{T}\boldsymbol{u}(t) + \boldsymbol{\tau}_{sd}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{p}_{sd}). \tag{4.21}$$

4.6 Segment Distance Computation

Segments of the HeiMan model do not have any bounds or border that would prevent them of colliding or overlapping with each other. When wanting to synthesize human-like motions one therefore has to ensure that segments do not overlap. For a pair segments this can be prevented by employing a distance measure for the two segments and ensuring that this distance measure is always positive. By creating a list of segment pairs of the whole model one can ensure non-overlapping segments of the whole model.

When using time-continuous optimization techniques as done in this thesis one further has to ensure that this is distance measure is a continuously differentiable function of the state of the model, which is in our case described by the generalized positions $\boldsymbol{q}(t)$. One way to do this would be to use spheres attached to each segment and use their distances. I.e. let $\boldsymbol{c}_1(\boldsymbol{q}), \boldsymbol{c}_2(\boldsymbol{q}) \in \mathbb{R}^3$ be the centers in the global reference frame and $r_1, r_2 \in \mathbb{R}$ radii of the two spheres. Then the distance measure would be

$$d(\boldsymbol{q}) = ||\boldsymbol{c}_1(\boldsymbol{q}) - \boldsymbol{c}_2(\boldsymbol{q})||_2 - r_1 - r_2. \tag{4.22}$$

However spheres only coarsely describe the shape of segments as the Shank or Thigh and therefore are unsuitable. Instead we use collision capsules to approximate the shapes of the segments as used e.g. in robotics (Stasse et al., 2008). A capsule is a cylinder with two hemispheres on both ends and is parameterized by its length $l \in \mathbb{R}$ and radius $r \in \mathbb{R}$. It can be seen as the convex hull of line segment of length l with spheres of radius r at each end (Figure 4.6).

For two capsules C_1 and C_2 the distance between the two capsules $d(C_1, C_2)$ is defined as the distance of the two closest points on the line segment minus the radii of each of the capsules. For $d(C_1, C_2) < 0$ we speak of penetration. This function is differentiable as long as the penetration is greater than the smallest radius of the two capsules. In the latter case the distance is not defined.

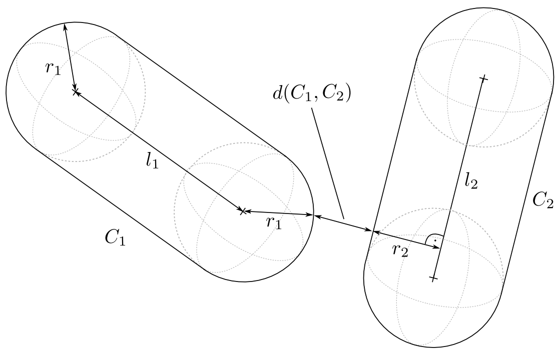


Figure 4.8: Schematic of collision capsules and the distance between them.

In our model we attach capsules to the frames of the approximating segments. The capsules are massless and therefore do not affect the physical properties but their position and orientation is dependent on the posture of the model and therefore the generalized positions $\boldsymbol{q}$. So for the segments *RightShank* and *LeftShank* we can ensure non-overlapping of these segments by the following constraint:

$$d(C_{RightShank}, C_{LeftShank}, \boldsymbol{q}) > 0 \tag{4.23}$$

4.7 Foot Contact Modeling

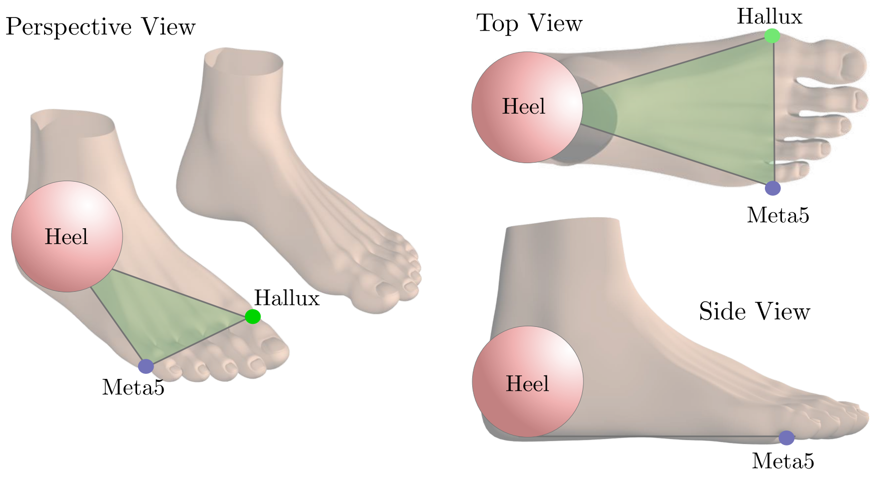


Figure 4.9: Schematic of the foot model used in this work. The sole of the foot is approximated by a triangle parallel to the ground when the model is standing in neutral posture.

During walking the human body is propelled by interaction of the feet with the ground. In a normal healthy gait the heel touches the ground first and rolls over the heel until the foot is flat on the ground. After this the heel lifts, so that only the frontal part of the foot and toes are in contact with the ground. At the end of the contact the foot then rolls over the toes and the foot is lifted and swings freely until the next heel strike. During contact the foot deforms due to the change of load of the body.

In our model we approximate the shape of the foot using a sphere and a line segment. The sphere is used for the heel contact and the line segment for the forefoot contact. The line segment is described by two points near the ball of the foot: *Hallux*, which is at the medial side and *Meta5*, which is at the lateral side near the distal end of the 5th Metatarsal bone. The heel sphere and the contact points are visualized in Figure 4.9. This foot model allows us to capture the essential types of contact during walking:

- **Heel Contact**: rolling contact of the heel sphere that starts with the heel strike and lasts until both the Hallux and 5th Metatarsal point are on the ground
- **Flat Contact**: both the heel sphere and the two contact points are in contact with the ground. This phase lasts until the heel lifts off.
- **Forefoot Contact**: this contact constrains the movement of the foot to the rotation around the axis that passes through the Hallux and Meta5 point. It ends when the foot is lifted from the ground.

Non-slipping Rolling Contact

The heel sphere is allowed to roll over the ground but slipping is not allowed. The rolling is modeled by constraining the tangential movement of the bottom point and the vertical movement of the sphere's center to zero. The constraints on the bottom point ensure that there is no slipping and the third constraint causes the rotation to be about the sphere's center. One should note that the bottom point moves on the surface of the sphere and its coordinates have to be recomputed at every time step.

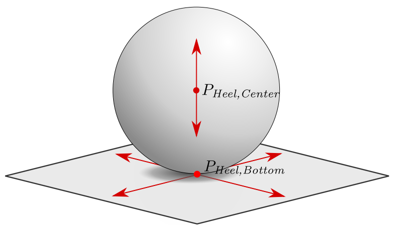


Figure 4.10: Acceleration constraints posed for non-slipping rolling contact. The bottom point of the sphere P_B is required to have zero acceleration along the tangential directions of the contact surface, whereas the sphere center P_C must not accelerate along the surface's normal direction.

The heel sphere is parameterized using the HeiMan parameters listed in Table 4.3. The center of the right heel is located in the coordinate frame of the right heel at $\boldsymbol{p}_{FootRightPosteriorPoint}$ and has a radius of $\boldsymbol{p}_{FootRightHeight}$. The parameters of the left heel sphere are described analogously.

The different types of contact are modeled using constraint sets introduced in Section 3.5.3. The individual constraint sets are described in the following paragraphs.

Constraint Set: Heel Contact

The constraint set for the contact during *heel contact* describes a non-slipping rolling contact. It is described by three translational constraints. Two of them are located at the sphere's contact point at the bottom of the sphere and act along two orthogonal tangential directions of the contact surface. The third constraint is specified at the sphere's center point and acts along the surface's normal direction. The constraint function for the heel contact can thus be written as:

$$\boldsymbol{g}_{Heel}(\boldsymbol{q}(t)) = \begin{bmatrix} g_{Heel,Bottom,X}(\boldsymbol{q}(t)) \\ g_{Heel,Bottom,Y}(\boldsymbol{q}(t)) \\ g_{Heel,Center,Z}(\boldsymbol{q}(t)) \end{bmatrix} : \mathbb{R}^{n_{dof}} \to \mathbb{R}^3,$$

where $g_{Heel,Bottom,X}(\boldsymbol{q}(t)) \in \mathbb{R}$ is the X–coordinate of the bottom point of the heel sphere in the global reference frame for the pose $\boldsymbol{q}(t) \in \mathbb{R}^{n_{dof}}$ at the time instant t. Similarly the $Y-$ and $Z-$ coordinates for the bottom point and center of the heel sphere are denoted by the remaining entries of $\boldsymbol{g}_{Heel}(\boldsymbol{q}(t))$.

Constraint Set: Flat Contact

For the constraint set of the flat contact one has to take care not to over-constrain the foot segment during *flat contact*. We use the following constraints: three translational constraints

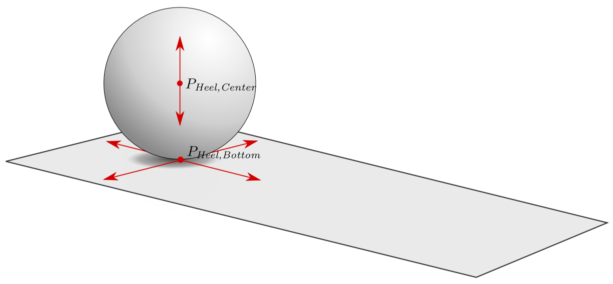


Figure 4.11: Contact constraints during heel contact.

for the heel sphere, two translational constraints at the 5th Metatarsal point along the global Y–axis (i.e. to the side) and the ground normal, and one translational constraint at the Hallux point along the ground normal. Using the same notation as in the previous constraint set we express this constraint set by:

$$\boldsymbol{g}_{Flat}(\boldsymbol{q}(t)) = \begin{bmatrix} g_{Heel,Bottom,X}(\boldsymbol{q}(t)) \\ g_{Heel,Bottom,Y}(\boldsymbol{q}(t)) \\ g_{Heel,Center,Z}(\boldsymbol{q}(t)) \\ g_{Meta5,Y}(\boldsymbol{q}(t)) \\ g_{Meta5,Z}(\boldsymbol{q}(t)) \\ g_{Hallux,Z}(\boldsymbol{q}(t)) \end{bmatrix} : \mathbb{R}^{n_{dof}} \rightarrow \mathbb{R}^6$$

.

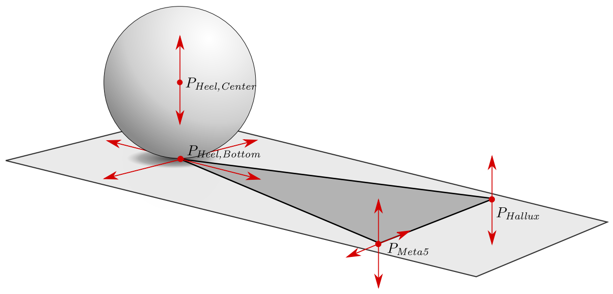


Figure 4.12: Contact constraints during flat contact. The dark grey triangular area represents the support polygon.

Constraint Set: Forefoot Contact

The constraint set for the forefoot contact consists of three translational constraints on the Hallux point along the three cordinate axes and two translational constraints on the 5th Metatarsal along the global X– and Z–axis. This is equivalent to a hinge joint that connects the forefoot to the ground and only allows a rotation around the axis passing through the Hallux and 5th Metatarsal points. It allows the model to excert a moment about the global Z–axis and therefore propel the body forward. This constraint set is expressed by:

$$\boldsymbol{g}_{Forefoot}(\boldsymbol{q}(t)) = \begin{bmatrix} g_{Hallux,X}(\boldsymbol{q}(t)) \\ g_{Hallux,Y}(\boldsymbol{q}(t)) \\ g_{Hallux,Z}(\boldsymbol{q}(t)) \\ g_{Meta5,X}(\boldsymbol{q}(t)) \\ g_{Meta5,Z}(\boldsymbol{q}(t)) \end{bmatrix} : \mathbb{R}^{n_{dof}} \rightarrow \mathbb{R}^5$$

.

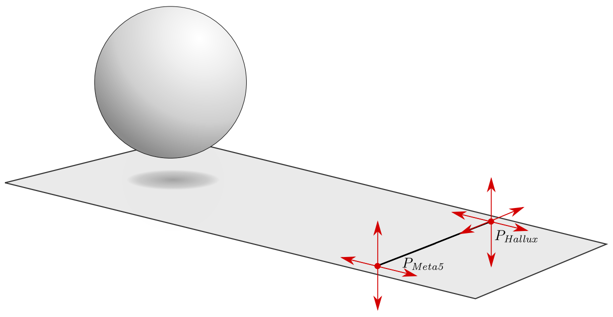


Figure 4.13: Contact constraints of the forefoot contact. The foot is only allowed to rotate around the line passing through P_{Hallux} and $P_{Metatarsal}$.

Double Support

One should note here that these constraint sets are only described for a single foot. However, there are phases during walking, in which more than one foot is in contact with the ground. In that case we use constraint sets that include constraints for both feet. E.g. when the right forefoot and the left heel is in contact with the ground we have a constraint set with eight constraints: five for the right forefoot and three for the left heel.

4.8 Constrained Whole-Body Dynamics as Ordinary Differential Equations

For $n_C \in \mathbb{N}$ distinct constraint sets we can express the dynamics of the actuated model under the presence of contact constraints as ODEs using:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{4.24}$$

with

$$\boldsymbol{x}(t) = \begin{bmatrix} \boldsymbol{q}(t) \\ \dot{\boldsymbol{q}}(t) \end{bmatrix} \in \mathbb{R}^{2n_{dof}} \tag{4.25}$$

being the differential state consisting of the generalized positions $\boldsymbol{q}(t) \in \mathbb{R}^{n_{dof}}$, generalized velocities $\dot{\boldsymbol{q}}(t) \in \mathbb{R}^{n_{dof}}$, actuations $\boldsymbol{u}(t) \in \mathbb{R}^{n_{actuated}}$, and $i \in 1, \ldots, n_C$ being the index of the active constraint set.

The constraints in constraint set i, $\boldsymbol{q}(t)$, $\dot{\boldsymbol{q}}$, and $\boldsymbol{u}(t)$ are then used to construct and solve the contact dynamics system (3.7) from Section 3.1.2. The computed accelerations $\ddot{\boldsymbol{q}}(t)$ are then used as the lower part of the right-hand side of $\boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t))$.

Chapter 5

Optimal Control Problems for Modeling Human Walking Motions

In this capter we describe the optimal control problem formulations that we developed to model human walking motions by means of a multi-phase optimal control problem formulation. We present two types of optimizations: a least-squares formulation that we use to reconstruct the dynamics of human movements from kinematic recordings and a second optimal control formulation with which we can generate a variety of walking movements.

We describe how we numerically solve the resulting infinite-dimensional optimal control problems using the *Direct Multiple-Shooting Method.* The resulting highly structured nonlinear optimization problem is described along with the Sequential Quadratic Programming (SQP) method that solves it.

5.1 Human Walking as a Multi-Phase Optimal Control Problem

In Chapter 1 we have described the human gait as a series of discrete phases. At each phase a different set of external contacts act on the human model. This results in a distinct ordinary differential equation (ODE) for each phase as described in Section 4.8. Furthermore contact collisions, such as the touchdown of the heel or forefoot, cause discontinuities in the joint velocities.

A multi-phase optimal control problem formulation therefore lends itself for the modeling of human walking as an optimization problem. For a given sequence of phases it can formally be defined as:

Definition 4. *Multi-Phase Optimal Control Problem with Discontinuities*

An optimal control problem with N phases over the time horizon $\mathcal{T} \overset{\text{def}}{=} [t_0, t_f] \subset \mathbb{R}$ on subintervals $\mathcal{T}_i = [t_{i-1}, t_i] \subset \mathcal{T}, i \in \mathcal{I} \overset{\text{def}}{=} \{1, \dots, N\}$, and $t_N = t_f$ is a constrained infinite-dimensional optimization problem that can be stated as:

$$\min_{\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{p},t_1,\dots,t_N} \quad \Phi(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) \tag{5.1a}$$

subject to:

$$\begin{aligned}
\dot{\boldsymbol{x}}(t) &= \boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}), & t \in \mathcal{T}_i, && (5.1\text{b})\\
\boldsymbol{x}(t_i^+) &= \boldsymbol{c}_i(\boldsymbol{x}(t_i^-)), & i \in \mathcal{I}, && (5.1\text{c})\\
\boldsymbol{g}_i(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) &\geq \boldsymbol{0}, & t \in \mathcal{T}_i, && (5.1\text{d})\\
\boldsymbol{r}(\boldsymbol{x}(\hat{t}_1), \dots, \boldsymbol{x}(\hat{t}_k), \boldsymbol{u}(\hat{t}_1), \dots, \boldsymbol{u}(\hat{t}_k), \boldsymbol{p}) &\geqq \boldsymbol{0}, & \hat{t}_1, \dots, \hat{t}_k \in \mathcal{T}, && (5.1\text{e})
\end{aligned}$$

in which we determine a piecewise continuous dynamic process $\boldsymbol{x}(t) : \mathcal{T} \to \mathbb{R}^{n_x}$, controls $\boldsymbol{u} : \mathcal{T} \to \mathbb{R}^{n_u}$, the time grid of phase switches $t_1, \ldots, t_N$, and a finite dimensional parameter vector $\boldsymbol{p} \in \mathbb{R}^{n_p}$ such that the objective function $\Phi : \mathcal{X} \times \mathcal{U} \times \mathbb{R}^{n_p} \to \mathbb{R}$ is minimized and the path constraints $\boldsymbol{g}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_g}, i \in \mathcal{I}$ and the equality and inequality point constraints $\boldsymbol{r} : (\mathbb{R}^{n_x})^k \times (\mathbb{R}^{n_u})^k \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_r}$ are fulfilled.

On each interval $\mathcal{T}_i$ the dynamics of the process are described by a set of ordinary differential equations with right hand sides $\boldsymbol{f}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ and the discontinuities of $\boldsymbol{x}(t)$ at t_i are described as transitions from the right-side limit t_i^- of $\mathcal{T}_i$ to the left-side limit t_i^+ of $\mathcal{T}_{i+1}$ using the phase transitions functions $\boldsymbol{c}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$.

$\mathcal{X} : \mathcal{T} \to \mathbb{R}^{n_x}$ is the set of all trajectories of the dynamics and $\mathcal{U} : \mathcal{T} \to \mathbb{R}^{n_u}$ the set of all control trajectories.

The objective function (5.1a) of an optimal control problem is commonly defined as

$$\Phi(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} \Phi_{L_i}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) dt + \sum_{i=1}^{N} \Phi_{M_i}(t_i, \boldsymbol{x}(t_i), \boldsymbol{p}), \tag{5.2}$$

which consists of the *Lagrange terms* $\Phi_{L_i} : \mathcal{T}_i \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}$ that are integrated over the phase subintervals $\mathcal{T}_i$ and *Mayer terms* $\Phi_{M_i} : \{t_i\} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}$ that are each evaluated at the end of a phase. Objective functions with both Lagrange and Mayer terms are also called *Bolza*-type objective functions. Lagrange-type and Mayer-type objective functions are equivalent in the sense that a Lagrange-type objective function can always be reformulated as a Mayer-type and vice versa. Nevertheless their explicit use eases the formulation of certain optimization criteria.

Each of the N right-hand sides (5.1b) describes the dynamics of the human model subject to a different set of contact constraints as an ODE that may be affected by the controls $\boldsymbol{u}(t)$ at any time. Furthermore the parameter vector $\boldsymbol{p}$ may affect the dynamic behaviour of the model. Throughout this chapter we will assume existence and uniqueness of the solution of the ODEs, i.e. that $\boldsymbol{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ is piecewise Lipschitz continuous.

Equations (5.1c) describe the transitions between phases, i.e. how the differential states change from the end of one phase to the beginning of the following phase. They are used to model contact collisions of the foot with the ground as instantaneous discontinuities in the differential states.

General path constraints are expressed using (5.1d). These include upper and lower bounds for controls and differential states.

In (5.1e) additional nonlinear point constraints are defined. These are used to formulate constraints such as foot clearance, proper contact force, periodicity, and others.

5.1.1 Numerical Methods for Solving Optimal Control Problems

In the following section we will describe various methods for the numerical solution of optimal control problems. We illustrate the essential concepts of these methods by considering a standard optimal control problem with a single phase and later discuss how the full problem (5.1) can be treated numerically. The standard optimal control problem is stated as:

Definition 5. *Standard Continuous Optimal Control Problem*

The simplified continuous optimal control problem is a constrained infinite-dimensional optimization problem on the time horizon $\mathcal{T} = [t_0, t_f] \subset \mathbb{R}$ that can be stated as:

$$\min_{\boldsymbol{x}(t), \boldsymbol{u}(t)} \quad \Phi_M(\boldsymbol{x}(t_f)) \tag{5.3a}$$

subject to:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), & t \in [t_0, t_f], \quad &(5.3b)\\ \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq \boldsymbol{0}, & t \in [t_0, t_f], \quad &(5.3c)\\ \boldsymbol{r}(\boldsymbol{x}(\hat{t}_0), \ldots, \boldsymbol{x}(\hat{t}_k)) &\geqq \boldsymbol{0}, & \hat{t}_1, \ldots, \hat{t}_k \in [t_0, t_f], \quad &(5.3d) \end{aligned}$$

in which we determine a piecewise continuous process $\boldsymbol{x}(t) : \mathcal{T} \to \mathbb{R}^{n_x}$ *and controls* $\boldsymbol{u}(t) : \mathcal{T} \to \mathbb{R}^{n_u}$ *such that the objective function* $\Phi : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ *is minimized and the path constraints* $\boldsymbol{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_g}$ *and point constraints* $\boldsymbol{r} : (\mathbb{R}^{n_x})^k \to \mathbb{R}^{n_r}$ *are fulfilled.*

The dynamic process is given as an ordinary differential equation with right-hand-side $\boldsymbol{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$.

This problem is only defined on a single phase on the fixed time interval $\mathcal{T} = [t_0, t_f] \subset \mathbb{R}$ and uses only a Mayer-type objective function (5.1.1). It furthermore does not include the free optimization parameter $\boldsymbol{p}$. The solution of the optimal control problem (5.3) are the infinite dimensional trajectories of the controls $\boldsymbol{u}(t)$ and differential states $\boldsymbol{x}(t)$.

One way to solve this infinite dimensional problem is the so-called *indirect* method. It uses Pontryagin's Maximum principle (Pontryagin, 1987) to transform (5.3) into a boundary value problem by formulating the so-called costate equations. The resulting boundary value problem can then be solved e.g. using the multiple-shooting method (Bock, 1981b; Hartl et al., 1995). For small systems this is a viable approach but quickly becomes complex as optimality conditions have to be derived by hand. Especially for the complex models used in this thesis this would not be possible as already the rigid multibody dynamics equations require automatic methods for the evaluation.

Instead we use a *direct* method to solve the optimal control problem. Direct methods approximate the infinite dimensional $\boldsymbol{u}(t)$ using a finite set of parameters, e.g. using piecewise constant functions. By further discretizing the differential states $\boldsymbol{x}(t)$ one obtains a nonlinear programming problem (NLP) that can then be solved using SQP. Common methods for state discretization include collocation, single-shooting, or a multiple-shooting approach.

Collocation methods discretize boundary value problems by first employing a grid on the time horizon. The resulting points are the so-called *collocation points*. Controls are then approximated between these points by a finite dimensional representation (e.g. piecewise constant functions), and the differential states are approximated by polynomials that are required to fulfill the differential equation (5.3b) on the collocation points. The resulting sparse nonlinear programming problem can then for example be solved using a structure exploiting SQP method. Knowledge or guesses of the optimal solution for the process can easily be integrated at the collocation points. A challenging aspect of collocation is the inclusion adaptive refinement of the collocation points to account for the nonlinearity in the ODEs as it is a priori unknown where a fine discretization is required. Furthermore this would lead to a change in the number of optimization variables of the NLP which would need further strategies for a proper treatment. A detailed description of direct collocation can be found in von Stryk and Bulirsch (1992) and is implemented in the numerical package DIRCOL (von Stryk, 2000).

The adaptivity problem for the process can easily be treated using single-shooting. It discretizes the process by a single state vector at the beginning of the time horizon to reduce it to a initial value problem. This initial value problem can then be solved using a standard ODE integrator with an adaptive stepsize control. The optimal control problem is then discretized by an NLP on the control parameters and the initial value of the process. A drawback of this approach is that prior knowledge of the optimal process can only be specified by the choice of initial value. Furthermore the ODE solver may fail to integrate the differential equation over the whole time horizon for stiff problems and an unsuitable choice of controls.

Multiple-shooting combines advantages of both collocation and single-shooting. It uses adaptive solvers for the ODEs and allows the incorporation of guesses for the optimal solution. Instead of performing a single integration over the whole time horizon it discretizes the time horizon using a finite grid and performs multiple integrations ("shootings") on each of the so-called *multiple-shooting intervals*. Knowledge of the optimal process can be used to initialize the differential states at the grid points, which are also referred to as *multiple-shooting nodes*. Multiple-shooting also lends itself for parallel computations as integrations on the individuals intervals can be performed independently of each other. Similar to collocation the resulting NLP is highly structured and should be exploited to achieve high performance. Direct multiple-shooting, together with descriptions on how to exploit the structures was published in Bock and Plitt (1984). A detailed description can also be found in Leineweber (1995).

5.2 The Direct Multiple-Shooting Method

In this work we use direct multiple-shooting as implemented in the software package MUSCOD-II (Leineweber et al., 2003), which is a C/C++ reimplementation of the original Fortran implementation of Bock et al. In the following we will describe its principles in more detail.

5.2.1 Discretization of the Optimal Control Problem

The optimal control problem (5.3) is written in terms of continuous functions $\boldsymbol{x}(t)$ and $\boldsymbol{u}(t)$ and is therefore also referred to as the *continuous* optimal control problem. Following the *"first discretize, then optimize"* approach of direct methods we will now describe how the continuous problem can be discretized.

Discretization of the Controls

To discretize the controls $\boldsymbol{u}(t)$ we first define a grid on the time horizon $\mathcal{I} = [t_0, t_f]$ with:

$$t_0 < t_1 < \ldots < t_{m-1} < t_m = t_f,\ I_j = [t_{j-1}, t_j],\ \text{for } j = 1, \ldots, m. \tag{5.4}$$

On each of these subintervals we replace the controls using a finite dimensional representation using base functions $\boldsymbol{\varphi}$ and parameters $\mathbf{q}_j$. We can then write the controls[1] as

$$\boldsymbol{u}(t) \approx \boldsymbol{u}(t; \mathbf{q}_{j-1}) = \boldsymbol{\varphi}(t, \mathbf{q}_{j-1}), \mathbf{q}_{j-1} \in \mathbb{R}^{k_u n_u}, t \in I_j = [t_{j-1}, t_j],\ \text{for } j = 1, \ldots, m \tag{5.5}$$

When using piecewise constant discretization we have $k_u = 1$ and $\boldsymbol{\varphi}(t, \mathbf{q}_j) = \mathbf{q}_j$ (see Figure 5.1). Discretizations using higher order base functions are also possible, e.g. for $k_u = 2$ we can discretize the controls using piecewise linear controls using the basis function

$$\boldsymbol{\varphi}(t, \mathbf{q}_j) = \mathbf{q}_j^1 + \frac{t - t_{j-1}}{t_j - t_{j-1}}(\mathbf{q}_j^2 - \mathbf{q}_j^1),\ \mathbf{q}_j = \begin{bmatrix} \mathbf{q}_j^1 \\ \mathbf{q}_j^2 \end{bmatrix} \in \mathbb{R}^{2n_u}. \tag{5.6}$$

We can obtain continuous piecewise linear controls by adding a constraint of the form $\mathbf{q}_j^2 - \mathbf{q}_{j+1}^1 = 0$ to the discretized optimal control problem. The discretized controls are denoted by $\boldsymbol{u}(t; \mathbf{q})$ and the vector $\mathbf{q} = (\mathbf{q}_0, \ldots, \mathbf{q}_{m-1})$ is called the vector of control parameters.

Control discretizations with higher order such as splines are also possible, however one has to keep in mind that this also increases the number of variables in the optimization. In this thesis we use continuous piecewise linear controls.

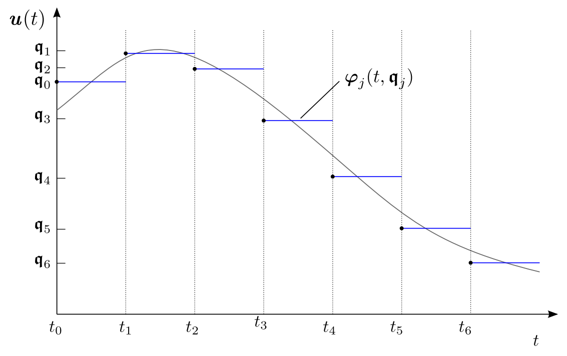


Figure 5.1: Piecewise constant control discretization of a scalar control function

[1]The $\mathbf{q}$ used here is not to be confused with the generalized positions of the differential states $\boldsymbol{q}$. The $\mathbf{q}$ in this section is solely used for the *control* discretization and has nothing to do with the *state* parameterization.

Parametrization of the Differential States using Multiple-Shooting

Multiple shooting parameterizes the states on a grid similarly to the control discretization seen above. For simplicity we assume the same grid for both control and state discretization, but it is important to note that the grids can be chosen independently.

The individual intervals are referred to as *multiple-shooting intervals* and the points $t_0, t_1, \ldots, t_m$ are called *multiple-shooting nodes.* At each node the differential states are specified by $\boldsymbol{s}_j \in \mathbb{R}^{n_x}$, that can be initialized e.g. using knowledge of the optimal process and/or interpolation.

Together with the control discretizations this allows defining m initial value problems (IVPs), one for each of these multiple-shooting intervals as

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{\varphi}(t, \mathbf{q}_{j-1})), \quad j = 1, \ldots, m, \ t \in I_j, && (5.7a) \\ \boldsymbol{x}(t_{j-1}) &= \boldsymbol{s}_{j-1}. && (5.7b) \end{aligned}$$

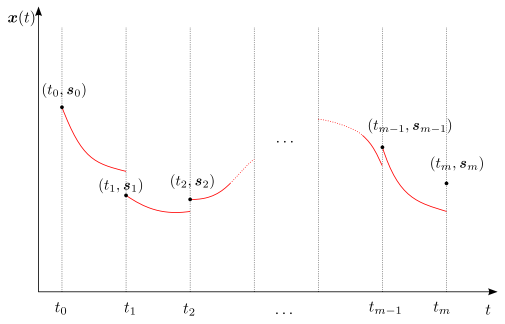
We denote the solution of the IVP starting at t_{j-1}, initial value $\boldsymbol{s}_{j-1}$, and discretized controls $\boldsymbol{\varphi}(\mathbf{q}_{j-1})$ evaluated at t with $\boldsymbol{\eta}(t; t_{j-1}, \boldsymbol{s}_{j-1}, \mathbf{q}_{j-1})$, or just $\boldsymbol{\eta}(t; t_{j-1}, \boldsymbol{s}_{j-1}, \mathbf{q}_{j-1})$.

The trajectories obtained by this are in general not continuous as the values obtained at the end point of the multiple-shooting interval do not coincide with the starting value of the next interval (see Figure 5.2). To obtain a continuous solution additional *continuity conditions* are employed:

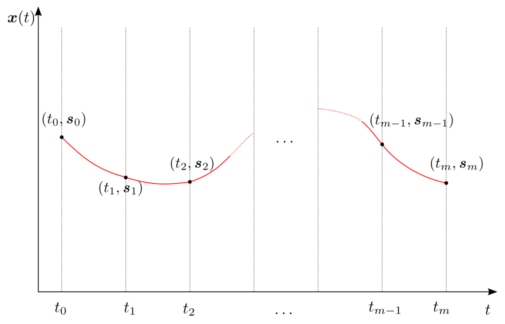
$$\boldsymbol{\eta}(t_j; t_{j-1}, \boldsymbol{s}_{j-1}, \mathbf{q}_{j-1}) - \boldsymbol{s}_j = \mathbf{0}, \text{ for } j = 1, \ldots, m, t \in I_j \tag{5.8}$$

By setting $\boldsymbol{s} = (\boldsymbol{s}_0, \ldots, \boldsymbol{s}_m)$ the continuous solution is characterized by the solution of following nonlinear problem:

$$\boldsymbol{h}(\boldsymbol{s}, \mathbf{q}) = \begin{bmatrix} \boldsymbol{\eta}(t_1; t_0, \boldsymbol{s}_0, \mathbf{q}_0) - \boldsymbol{s}_1 \\ \vdots \\ \boldsymbol{\eta}(t_{m-1}; t_{m-2}, \boldsymbol{s}_{m-2}, \mathbf{q}_{m-2}) - \boldsymbol{s}_{m-1} \\ \boldsymbol{\eta}(t_m; t_{m-1}, \boldsymbol{s}_{m-1}, \mathbf{q}_{m-1}) - \boldsymbol{s}_m \end{bmatrix} = \mathbf{0} \tag{5.9}$$



(a) Initial guess of the process where the continuity conditions are violated.



(b) Solution of (5.9) with fulfilled continuity conditions.

Figure 5.2: State discretization using the multiple-shooting method

Discretization of the Path Constraints

A simple approach to discretize the path constraints can be done by only evaluating them on the multiple-shooting nodes:

$$\boldsymbol{g}(\boldsymbol{s}_j, \mathbf{q}_j) \geqq \mathbf{0}, \text{ for } j = 1, \ldots, m-1 \tag{5.10}$$

More sophisticated methods to ensure feasibility of the differential states within the multiple-shooting intervals are described in Potschka et al. (2009). These are not used in this work however, as small violations of the constraints are acceptable for us.

Treatment of the Point Constraints

The point constraints (5.3d) can be treated by choosing the multiple-shooting grid such that it is a superset of the time values of the point constraints. I.e. for the point constraints

$$\boldsymbol{r}(\boldsymbol{x}(\hat{t}_0), \dots, \boldsymbol{x}(\hat{t}_k)) \geqq \boldsymbol{0},\ \hat{t}_1, \dots \hat{t}_k \in [t_0, t_f], \tag{5.11}$$

one has to choose $t_0, \dots, t_m$ such that

$$\{\hat{t}_0, \dots, \hat{t}_k\} \subseteq \{t_1, \dots, t_m\} \tag{5.12}$$

holds. The point constraints can then be evaluated using the state parameters $\boldsymbol{s}_j, j = 1, \dots, m$.

Free Parameters

It may be of use to not only optimize the controls $\boldsymbol{u}(t)$ and states $\boldsymbol{x}(t)$ but also a vector of free parameters $\boldsymbol{p} \in \mathbb{R}^{n_p}$. This vector may in our case contain the walk velocity, step length, unknown spring-damper constants, or other values that are constant over the step cycle. In this case $\boldsymbol{p}$ is simply added as a vector of free variables to the optimization problem and as argument to objective and constraint functions.

Free Phase Durations

The time horizon in (5.3) is on the fixed time horizon $[t_0, t_f] \subset \mathbb{R}$, however it can easily be extended such that the duration of the phase is a free optimization variable. To do so let $d = t_f - t_0 \in \mathbb{R}$ denote the duration of the time horizon. By employing a time transformation as

$$t(\tau) \overset{\text{def}}{=} t_0 + d\tau \tag{5.13}$$

one can write (5.3) with a free duration as:

$$\min_{\boldsymbol{x}(t), \boldsymbol{u}(t), d} \quad \Phi_M(\boldsymbol{x}(t(1))) \tag{5.14a}$$

subject to:

$$\begin{aligned}
\dot{\boldsymbol{x}}(\tau) &= \boldsymbol{f}(\boldsymbol{x}(t(\tau)), \boldsymbol{u}(t(\tau))), && \tau \in [0,1], && (5.14\text{b})\\
\boldsymbol{g}(\boldsymbol{x}(t(\tau)), \boldsymbol{u}(t(\tau))) &\geq \boldsymbol{0}, && \tau \in [0,1], && (5.14\text{c})\\
\boldsymbol{r}(\boldsymbol{x}(t(\hat{\tau}_0)), \dots, \boldsymbol{x}(t(\hat{\tau}_k))) &\geqq \boldsymbol{0}, && \hat{\tau}_1, \dots \hat{\tau}_k \in [0,1]. && (5.14\text{d})
\end{aligned}$$

Treatment of Multiple Phases and Discontinuities

The multiple-shooting discretization also allows formulating problems with multiple phases, i.e. different right-hand sides for the process as in (5.1b). For example, for a problem with two phases one can use the first m_l with $0 < m_l < m$ multiple-shooting intervals for the first phase and the remaining $m - m_l$ intervals for the second phase.

If the transition of the two phases is continuous one has the same continuity conditions (5.8) as at any other multiple-shooting node. To formulate discontinuities between phases that are described using the phase transition function $\boldsymbol{c}_1 : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ one has to replace the continuity condition with

$$\boldsymbol{c}_1(\boldsymbol{\eta}(t_{m_l}; t_{m_l-1}, \boldsymbol{s}_{m_l-1}, \mathbf{q}_{m_l-1})) - \boldsymbol{s}_{m_l} = \boldsymbol{0},\ t \in [t_{m_l-1}, t_{m_l}] \tag{5.15}$$

to properly take the phase transition from phase 1 to phase 2 into account.

5.2.2 Discretized Optimal Control Problem

By combining the discretized variables $\mathbf{q}$ and $\boldsymbol{s}$, the free parameters $\boldsymbol{p}$, and durations $d_1, \dots, d_N$ of the N phases into a vector

$$\boldsymbol{y} = (\boldsymbol{s}_0, \mathbf{q}_0, \dots, \boldsymbol{s}_{m-1}, \mathbf{q}_{m-1}, \boldsymbol{s}_m, \boldsymbol{p}, d_1, \dots, d_N) \tag{5.16}$$

and using the discretized constraints arising from the multiple-shooting discretization (5.9), discretized path constraints (5.10), and phase discontinuities (5.15) we can write the discretized optimal control problem as a nonlinear program:

$$\min_{\boldsymbol{y}} F(\boldsymbol{y}) = \Phi_M(\boldsymbol{s}_m) \tag{5.17a}$$

subject to:

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{y}) &= \mathbf{0} && (5.17b) \\ \boldsymbol{g}(\boldsymbol{y}) &\geq \mathbf{0} && (5.17c) \end{aligned}$$

The problem size in terms of number of variables and number of constraints depends on the application and choice of the discretization grid. In our case the number of variables is given by:

$$n_y = n_x(m+1) + n_u k_u m + n_p + N. \tag{5.18}$$

5.2.3 Solution of the Discretized Optimal Control Problem

In principle one could directly apply a general nonlinear programming code, such as Sequential Quadratic Programming (SQP) to solve (5.17). SQP methods can be motivated by applying the Newton method on the first-order necessary conditions (the so-called *Karush-Kuhn-Tucker* (KKT) conditions) for optimality of nonlinear constrained optimization problems. For an in-depth description of SQP methods we refer to Nocedal and Wright (2006) or Gill et al. (1981).

In the following we will focus our discussion on the most important aspects of SQP in combination with direct multiple-shooting. SQP solves nonlinear programs, such as (5.17) iteratively, i.e. by generating a sequence

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \alpha \boldsymbol{\delta y}_k, \tag{5.19}$$

where $\boldsymbol{\delta y}_k$ is the step direction and $\alpha > 0 \in \mathbb{R}$ the step length. The latter is obtained by employing a line search method as a globalization strategy.

The step direction $\boldsymbol{\delta y}_k$ is obtained by solving a quadratic problem that approximates (5.17) and can be stated as:

$$\min_{\boldsymbol{\delta y}_k} \nabla \boldsymbol{F}(\boldsymbol{y}_k)^T \boldsymbol{\delta y}_k + \frac{1}{2} \boldsymbol{\delta y}_k^T \boldsymbol{B}_k \boldsymbol{\delta y}_k \tag{5.20a}$$

subject to:

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{y}_k) + \nabla \boldsymbol{h}(\boldsymbol{y}_k)^T \boldsymbol{\delta y}_k &= \mathbf{0}, && (5.20b) \\ \boldsymbol{g}(\boldsymbol{y}_k) + \nabla \boldsymbol{g}(\boldsymbol{y}_k)^T \boldsymbol{\delta y}_k &\geq \mathbf{0}, && (5.20c) \end{aligned}$$

where $\boldsymbol{B}_k$ is the Hessian of the Lagrangian. In practice $\boldsymbol{B}_k$ is often approximated, e.g. by using the BFGS update method. For least-squares objective functionals a good choice would be a Gauss-Newton approximation.

The Hessians $\boldsymbol{B}_k$ and Jacobians $\nabla \boldsymbol{h}(\boldsymbol{y}_k)$ and $\nabla \boldsymbol{g}(\boldsymbol{y}_k)$ are sparse and highly structured. These sparsities are due to the direct multiple-shooting approach. One example are the continuity conditions that are part of (5.17b) only: they are only dependent on the discretized controls and differential state parameters of two consecutive multiple-shooting nodes and the free parameter vector $\boldsymbol{p}$. As a result the remaining entries in the rows of the Jacobian are zero.

These structures can be exploited using the so-called *condensing* method, which was described in Bock and Plitt (1984). It removes the sparsities that are caused by multiple-shooting and results in a much smaller and dense quadratic problem that can be solved with a standard QP solver, e.g. QPOPT (Gill et al., 1995).

One important aspect is also the evaluation Jacobians which contain sensitivity information of the process, i.e. the derivatives of the integrated values $\boldsymbol{\eta}(t_j; t_{j-1}, \boldsymbol{s}_{j-1})$ with respect to the initial values $\boldsymbol{s}_j$. A naïve approach would treat the numerical integration of the initial value problems (5.7a) as a black box function evaluation and perform finite differences around it. However this approach, which is also called *External Numerical Differentiation* (END) ignores the fact that different starting values may result in different step sizes in the adaptive integrator.

Internal Numerical Differentiation (IND) on the other hand uses the same integrator steps for both nominal and perturbed trajectories to compute the operands of the finite difference computation. As such it requires special numerical integrators to allow this. IND was originally proposed in Bock (1981a) and yields higher accuracy in the sensitivities and ensures that they are consistent with the numerical solutions of the nominal trajectories. This allows computing accurate sensitivities with fewer function evaluations without having to provide analytic derivatives. It results in a significant performance increase of the solution of the OCP as sensitivity generation is the most expensive part for the type of problems used in this thesis.

5.3 Dynamics Reconstruction Using a Least-Squares Formulation

We now describe the least-squares optimization problem that reproduces the fitted motion capture data from Section 4.4.3 using the subject-specific dynamic model. Whereas the fitting in Chapter 4 was only using a kinematic model, the optimizations performed here consider the dynamics of a full-body model. The computations here are much more expensive, however, the use of numerical optimal control allows us to reconstruct joint actuations that are otherwise unknown.

The aim of these optimizations are twofold: first we are interested in the joint actuations that produce a given motion and second the result of these optimizations give insight into how well the dynamic model can actually reproduce the motions of the motion capture data.

The formulation presented here performs a least-squares fit on a single step, i.e. it starts right after the left foot lifts off the ground and ends when the right foot lifts off the ground. This is done to reduce the amount of computations and is sufficient as the gaits of interest are near symmetric. However, the formulation presented here can be extended to double-step or multi-step sequences to treat asymmetries in the gait.

5.3.1 Problem Formulation

The least-squares optimal control problem approximates a given recorded motion using the dynamic subject-specific model. Let the recorded motion capture data be given as a set of time discrete postures in form of generalized positions $\boldsymbol{q}_j^D \in \mathbb{R}^{n_{dof}}, j = 0, \dots, m$ at time points $t_0, \dots t_m \in \mathbb{R}$. Then the dynamics reconstruction problem can be described by

$$\min_{\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)} \quad \sum_{j=0}^{m} \frac{1}{2} \left(||\boldsymbol{q}_j^D - \boldsymbol{q}(t_j)||_2^2 + \gamma_u || \boldsymbol{W} \boldsymbol{u}(t_j)||_2^2 \right) \tag{5.21a}$$

subject to:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}_i(\boldsymbol{x}(t), \boldsymbol{u}(t)), & t \in [t_{i-1}, t_i], i \in \mathcal{I}, && (5.21b)\\ \boldsymbol{x}(t_i^+) &= \boldsymbol{c}_i(\boldsymbol{x}(t_i^-)), & i \in \mathcal{I}, && (5.21c)\\ \boldsymbol{g}_i(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geqq \boldsymbol{0}, & t \in [t_{i-1}, t_i], i \in \mathcal{I}, && (5.21d)\\ \boldsymbol{r}(\boldsymbol{x}(\hat{t}_1), \dots, \boldsymbol{x}(\hat{t}_k), \boldsymbol{u}(\hat{t}_1), \dots, \boldsymbol{u}(\hat{t}_k)) &\geqq \boldsymbol{0}, & \hat{t}_1, \dots \hat{t}_k, \in \mathcal{T}, && (5.21e) \end{aligned}$$

where $\boldsymbol{x}(t)$ are the differential states with

$$\boldsymbol{x}(t) = \begin{bmatrix} \boldsymbol{q}(t) \\ \dot{\boldsymbol{q}}(t) \end{bmatrix} \in \mathbb{R}^{2n_{dof}}, \tag{5.22}$$

and where $\boldsymbol{q}(t) \in \mathbb{R}^{n_{dof}}$ are the generalized position variables and $\dot{\boldsymbol{q}}(t) \in \mathbb{R}^{n_{dof}}$ the generalized velocity variables of the model. The controls $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$, with $n_u = n_{actuated}$ are the actuations in form of joint torques that are acting directly at the joints of the model.

The dynamics of the model during the different phases are described using the ordinary differential equations (5.1b). The individual right-hand side functions $\boldsymbol{f}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ describe the different dynamics subject to the different constraint sets (see Section 4.8) and are influenced by the controls $\boldsymbol{u} : \mathcal{T} \to \mathbb{R}^{n_u}$.

The optimization computes optimal state trajectories $\boldsymbol{x}(t) = [\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)]^T$ and joint actuations $\boldsymbol{u}(t)$. The duration of the individual phases are prescribed from the motion capture data and there are no free parameters.

Phase transitions, specifically collisions at touch-down events of the foot with the ground, are described using the phase transition functions $\boldsymbol{c}_i(\boldsymbol{x}(t))$. The collisions are evaluated as described in Section 3.1.3 with at restitution parameter $e = 0$.

The path constraints (5.21d) contain upper and lower bounds for the differential states $\boldsymbol{x}(t)$ and controls $\boldsymbol{u}(t)$. Upper and lower bounds for the generalized coordinates $\boldsymbol{q}(t)$ are shown in Table 5.1. The bounds are obtained from the motion capture data by using the upper and lower bounds across all recorded emotions and an offset by a margin such that the motion is not constrained by these bounds in the final solution. For the generalized velocities $\dot{\boldsymbol{q}}(t)$ we used the bounds $-15.0 \leq \dot{q}_i(t) \leq 15.0, i = 1, \dots, n_{dof}$. For the controls $\boldsymbol{u}(t)$ we use the bounds listed in Table 5.2 that were chosen such that they are not reached in the solutions.

Joint	Axis	Min	Max	Joint	Axis	Min	Max
Pelvis	TX	-0.11	1.14	*AnkleLeft*	RY	-0.45	0.46
	TY	-0.10	0.10		RX	-0.25	0.20
	TZ	0.80	1.05		RZ	-0.43	0.45
	RY	-0.20	0.20	*MiddleTrunk*	RY	-0.20	0.33
	RX	-0.24	0.28		RX	-0.39	0.43
	RZ	-0.50	0.37		RZ	-0.28	0.33
HipRight	RY	-0.88	0.65	*ShoulderRight*	RY	-0.57	1.20
	RX	-0.38	0.30		RX	-0.55	-0.14
	RZ	-0.39	0.30		RZ	-0.35	0.90
KneeRight	RY	-0.01	0.95	*ElbowRight*	RY	-1.75	0.06
AnkleRight	RY	-0.45	0.35	*ShoulderRight*	RY	-0.60	0.79
	RX	-0.18	0.35		RX	0.03	0.29
	RZ	-0.54	0.20		RZ	-0.90	0.42
HipLeft	RY	-0.79	0.73	*ElbowRight*	RY	-1.65	0.39
	RX	-0.30	0.35	*Neck*	RY	-0.25	0.85
	RZ	-0.30	0.95		RX	0.02	0.26
KneeLeft	RY	-0.01	1.67		RZ	-0.24	0.33

Table 5.1: Upper and lower bounds of the generalized positions $\boldsymbol{q}(t)$. Axes with a T in the name denote a translational and with a R a rotational degree of freedom.

The interior point constraints of (5.21e) contain additional constraints such as proper ground reaction forces and kinematic constraints (e.g. the foot should rest at the beginning of the step). These constraints will be described in detail in Section 5.3.3.

5.3.2 Objective Function

The first term in the objective function (5.21a) minimizes the sum of squared errors of the joint angles of the model $\boldsymbol{q}(t_k)$ with the joint angles from the motion capture data $\boldsymbol{q}_k^D$.

The second term of the objective function is used to regularize the problem and minimizes the applied joint forces weighted by the diagonal matrix $\boldsymbol{W} = diag\{w_l\}, w_l > 0, l = 1, \dots, n_u$.

Joint	Axis	Min	Max	Joint	Axis	Min	Max
HipRight	*RY*	-200	200	*MiddleTrunk*	*RY*	-100	100
	RX	-100	100		*RX*	-100	100
	RZ	-100	100		*RZ*	-100	100
KneeRight	*RY*	-200	100	*ShoulderRight*	*RY*	-50	50
AnkleRight	*RY*	-100	300		*RX*	-50	50
	RX	-150	100		*RZ*	-50	50
	RZ	-100	100	*ElbowRight*	*RY*	-50	50
HipLeft	*RY*	-100	100	*ShoulderRight*	*RY*	-50	50
	RX	-100	100		*RX*	-50	50
	RZ	-100	100		*RZ*	-50	50
KneeLeft	*RY*	-200	200	*ElbowRight*	*RY*	-50	50
AnkleLeft	*RY*	-100	100	*Neck*	*RY*	-50	50
	RX	-100	100		*RX*	-50	50
	RZ	-100	100		*RZ*	-50	50

Table 5.2: Upper and lower bounds of the controls $\boldsymbol{u}(t)$.

This weighting is required to account for the different magnitudes of joint forces. The second term of the objective function is furthermore scaled by the factor $\gamma_u = 1.0 \times 10^{-2}$ to ensure that the joint residuals are the major contributor in the objective function.

The least-squares objective function gives rise to special structures in its derivatives. We exploit this using a Gauss-Newton approximation for the Hessian matrix in the discretized optimal control problem during the numerical solution.

As the motion capture data is recorded at a fixed sampling rate it may be that the time of one or more recorded frames do not coincide with those of the multiple-shooting nodes. One way to treat this would be to restrict the choice of multiple-shooting nodes such that they coincide or are a subset of the recorded motion capture frames.

To circumvent this restriction we interpolate the motion capture data and use the interpolated values as values of the motion capture joint angles at the multiple-shooting nodes. For this we used Catmull-Rom splines (Barry and Goldman, 1988). The index k on the data values $\boldsymbol{q}_k^D$ denote the joint angles of the interpolated motion capture data at the time values t_k.

5.3.3 Phase Descriptions

In the following we will describe the nonlinear point constraints for each phase. These constraints are used to formulate proper ground contact of the model with the ground and also to define the geometric events that describe the switching condition to describe the transition from one phase to another. We employ the following notation to describe position, velocity, and contact force acting at a contact point along a certain coordinate axes:

- $P_C^{S,N}(\boldsymbol{x}(t)) \in \mathbb{R}$ denotes the coordinate C of the position of contact point N on side S of the model,
- $V_C^{S,N}(\boldsymbol{x}(t)) \in \mathbb{R}$ denotes the coordinate C of the velocity of contact point N on side S of the model,
- and $F_C^{S,N}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \in \mathbb{R}$ denotes the coordinate C of the ground reaction force of contact point N on side S of the model,

where $S \in \{$*Right, Left*$\}$, denotes the side, i.e. whether the point is specified for the left or right foot, $N \in \{$*Heel, Hallux, Meta5*$\}$ denotes the contact point, and $C \in \{X, Y, Z\}$ the direction in the global coordinate frame along which the contact constraint is defined. The individual points *Heel, Hallux, Meta5* are described in Section 4.7 and an illustration of their locations is shown in Figure 4.9.

Phase 1: *Right Flat*

The following constraints are posed at the beginning of the *Right Flat* phase:

$$
\begin{aligned}
P_Z^{Right,Heel}(\boldsymbol{x}(t_0)) &= 0,\\
P_Z^{Right,Hallux}(\boldsymbol{x}(t_0)) &= 0,\\
P_Z^{Right,Meta5}(\boldsymbol{x}(t_0)) &= 0,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t_0)) &= 0,\\
V_X^{Right,Heel}(\boldsymbol{x}(t_0)) &= 0,\\
V_Y^{Right,Heel}(\boldsymbol{x}(t_0)) &= 0,\\
V_Z^{Right,Heel}(\boldsymbol{x}(t_0)) &= 0,\\
V_Y^{Right,Hallux}(\boldsymbol{x}(t_0)) &= 0,\\
V_Z^{Right,Hallux}(\boldsymbol{x}(t_0)) &= 0,\\
V_Z^{Right,Meta5}(\boldsymbol{x}(t_0)) &= 0,\\
F_Z^{Right,Heel}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0.
\end{aligned}
$$

The first four constraints ensure that the right foot is flat on the ground and the left foot is touching the ground with the Hallux point. The remaining constraints are used to fulfill the invariants of the algebraic constraint equation on velocity level.

Within the phase we have:

$$
\begin{aligned}
F_Z^{Right,Heel}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1
\end{aligned}
$$

i.e. we have positive ground reaction forces on all three points of our foot model. These are required to model a unilateral constraint. Without these the model exerts forces that would pull it towards the ground.

Phase 2: *Right Forefoot*

At the beginning of the second phase the following constraints have to be fulfilled:

$$
\begin{aligned}
F_Z^{Right,Heel}(\boldsymbol{x}(t_1), \boldsymbol{u}(t_1)) &= 0,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_1), \boldsymbol{u}(t_1)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_1), \boldsymbol{u}(t_1)) &\geq 0,\\
P_Z^{Left,Heel}(\boldsymbol{x}(t_1)) &\geq 0,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t_1)) &\geq 0
\end{aligned}
$$

First, the constraint forces on the right heel must have vanished while the constraint force on the right Hallux point is still positive. Furthermore the left Heel and Hallux points must not penetrate the ground.

During the phase the constraints are:

$$\begin{aligned}
P_Z^{Left,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_2,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_2.
\end{aligned}$$

The first three constraints enforce that the left Heel and Hallux points and the right Heel are not below the ground. The last two constraints ensure that the ground reaction force on the fore foot is positive.

Transition 1: *Collision Left Heel*

The first discontinuity occurs while only the right foot is in contact with the ground and the left heel touches down. This is described by the constraints

$$\begin{aligned}
P_Z^{Left,Heel}(\boldsymbol{x}(t_2)) &= 0,\\
V_Z^{Left,Heel}(\boldsymbol{x}(t_2)) &\leq 0,\\
P_Z^{Left,Meta5}(\boldsymbol{x}(t_2)) &\geq 0,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t_2)) &\geq 0,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t_2)) &\geq 0,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0.
\end{aligned}$$

The first equation ensures that the left heel is on ground level while the second confirms the touch-down. Without the confirmation it could be that the foot touches the ground but only moves parallel to the ground or even swinging up again. The remaining three equations enforce that the other foot points are not below the ground.

One should note that in this transition "phase" no integration is performed. Instead only the discontinuity in the generalized velocities $\dot{\boldsymbol{q}}$ due to the touch down collision is computed (see Section 3.1.3).

Phase 3: *Right Forefoot Left Heel*

After the transition the phase *Right Forefoot Left Heel* starts. At the beginning and during this phase we require:

$$\begin{aligned}
P_Z^{Left,Hallux}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_3,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_3,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3,\\
F_Z^{Left,Heel}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3.
\end{aligned}$$

The left Hallux point and the right Heel point should not be below the ground and the ground reaction forces on the two contact points right Hallux and left heel should be of proper sign.

Transition 2: ***Collision Left Forefoot***

The second discontinuity occurs when the left forefoot touches down. We formulate this using:

$$\begin{aligned} P_Z^{Left,Hallux}(\boldsymbol{x}(t_3)) &= 0, \\ P_Z^{Left,Meta5}(\boldsymbol{x}(t_3)) &= 0, \\ V_Z^{Left,Hallux}(\boldsymbol{x}(t_3)) &\leq 0, \\ V_Z^{Left,Meta5}(\boldsymbol{x}(t_3)) &\leq 0, \\ P_Z^{Right,Heel}(\boldsymbol{x}(t_3)) &\geq 0, \\ F_Z^{Right,Hallux}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0, \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0, \\ F_Z^{Left,Heel}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0. \end{aligned}$$

The first four equation describe the touch-down of the forefoot analogously as in the previous transition phase: the contact points are on ground level and have a negative contact velocity (i.e. a collision is imminent). The additional fifth constraint ensures that the right heel is not below the ground.

Again, there is no integration in this phase and the discontinuities in the generalized velocities is updated such that the contact velocities are set to zero.

Phase 4: ***Right Forefoot Left Flat***

At the beginning of this phase the left foot is flat on the ground and does not move due to the transition phase. During this final phase the weight is completely transferred onto the left foot. At the beginning and during this phase we have the constraints

$$\begin{aligned} P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_4, \\ F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4, \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4, \\ F_Z^{Left,Heel}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4, \\ F_Z^{Left,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4, \\ F_Z^{Left,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4. \end{aligned}$$

The right heel must not be below the ground and the ground reaction forces for all contact points must point upwards.

At the end of the phase we have a separate set of constraints that allow the right foot to be lifted from the ground:

$$\begin{aligned} F_Z^{Right,Hallux}(\boldsymbol{x}(t_4), \boldsymbol{u}(t_4)) &= 0, \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t_4), \boldsymbol{u}(t_4)) &= 0, \\ P_Z^{Right,Heel}(\boldsymbol{x}(t_4)) &\geq 0, \\ F_Z^{Left,Heel}(\boldsymbol{x}(t_4), \boldsymbol{u}(t_4)) &\geq 0, \\ F_Z^{Left,Hallux}(\boldsymbol{x}(t_4), \boldsymbol{u}(t_4)) &\geq 0, \\ F_Z^{Left,Meta5}(\boldsymbol{x}(t_4), \boldsymbol{u}(t_4)) &\geq 0. \end{aligned}$$

The normal contact force of the right Hallux point must vanish (first equation) while the right heel point is at least on ground level and all other contact points must have a position ground reaction force along the Z–axis.

5.3.4 Initialization

As the SQP-method is essentially a Newton-type method we cannot expect the problem to quickly converge to a solution from an arbitrary initial guess. For the least-squares optimization a natural choice for the initialization are the reference data of the joint angles that we obtained using the kinematic fit described in Section 4.4.3. However, this fitting data only contains positional information on a discrete time grid that may not coincide with the multiple-shooting grid.

Instead we use the values $\boldsymbol{q}_k^D$ obtained from Catmull-Rom spline interpolation that we also use for the least-squares objective function. Using spline interpolation has the advantage that we can initialize both joint positions $\boldsymbol{q}(t)$ and joint velocities $\dot{\boldsymbol{q}}(t)$ at all multiple-shooting nodes and ensure that they are consistent with each other. As initial values for the controls we use $\boldsymbol{u}(t) = \boldsymbol{0}$.

We use fixed durations of the individual phases. The durations of the individual phases were determined by visually inspecting the original motion capture data for contact gain and lift-off events.

5.4 Gait Synthesis Formulation

Our *Gait Synthesis Formulation* generates walking motions and actuations for a dynamic model that are optimal to a given objective function. This problem is more difficult to formulate as the least-squares optimization presented in the previous section as the latter automatically avoids non-physical artefacts such as penetration of limbs as the input motion is already valid in that respect. Furthermore both objective function and constraints must be carefully formulated when creating proper walking motions. E.g. simply minimizing the expended energy would lead to no movement at all if no constraint or an additional term in the objective prevents this.

Specifically the following properties are important aspects of this formulation:

Periodic Symmetry The motion we describe using this formulation is assumed to be periodic and symmetric. It is *periodic* in the sense that after a double step the posture of the model is equal to that of the beginning of the double step and translated along the walking direction. It is furthermore *symmetric* such that the posture after the first step mirrored at the sagittal plane is the same as that of the beginning of the step. This allows us to focus our efforts on the formulation of a single step which can then be mirrored to produce a full double step. The formulation can however also be extended to allow for asymmetric or non-periodic gaits. We use symmetric periodicity for joint positions $\boldsymbol{q}(t)$, joint velocities $\dot{\boldsymbol{q}}(t)$, and controls $\boldsymbol{u}(t)$.

Active and Passive Actuation The previous optimization formulation focused on the reconstruction of unknown forces $\boldsymbol{u}(t)$ acting on the joints during walking. These forces are produced by active and passive components of the human locomotion system. Active forces are produced by the muscles whereas the passive forces come from tendons and ligaments.

In this gait synthesis formulation the passive forces play an important role in the optimized forces as simply minimizing the cumulative joint forces would minimize the mechanical effort. Passive elements may also support the efficiency as they can be exploited by the active actuation. To model this effect we use spring dampers as described in Section 4.5 to model these passive elements. The parameters for these (i.e. spring constant, damping coefficient, rest position) can either be prescribed or added as free parameters $\boldsymbol{p}$ to the optimization problem to identify optimal values for them.

Segment Overlap Prevention In this optimization we further have prevent overlapping of segments as the optimization otherwise may produce unrealistic movements in which segments penetrate each other. To do so we use bounding capsules as described in Section 4.6 to model distances between segments. Instead of enforcing non-overlapping over all segment pairs we found it sufficient to enforce it between the two shank and the two foot segments.

In Table 5.3 we list the upper and lower bounds for the generalized positions $\boldsymbol{q}(t)$. They are greater or equal to the bounds in the previous problem and the additional range allows for more variations in the motions. For both $\dot{\boldsymbol{q}}(t)$ and $\boldsymbol{u}(t)$ we use the same bounds as in the reconstruction problem.

Joint	Axis	Min	Max	Joint	Axis	Min	Max
Pelvis	TX	0.00	1.14	*AnkleLeft*	RY	-0.45	0.46
	TY	-0.10	0.10		RX	-0.25	0.17
	TZ	0.87	0.98		RZ	-0.22	0.22
	RY	-0.20	0.20	*MiddleTrunk*	RY	-0.20	0.23
	RX	-0.24	0.23		RX	-0.39	0.33
	RZ	-0.32	0.27		RZ	-0.28	0.30
HipRight	RY	-0.75	0.57	*ShoulderRight*	RY	-0.57	0.89
	RX	-0.30	0.30		RX	-0.35	-0.14
	RZ	-0.39	0.30		RZ	-0.35	0.90
KneeRight	RY	-0.01	0.90	*ElbowRight*	RY	-1.40	-0.03
AnkleRight	RY	-0.45	0.35	*ShoulderRight*	RY	-0.51	0.89
	RX	-0.18	0.25		RX	0.03	0.29
	RZ	-0.22	0.22		RZ	-0.90	0.42
HipLeft	RY	-0.72	0.63	*ElbowRight*	RY	-1.40	-0.03
	RX	-0.30	0.29	*Neck*	RY	-0.25	0.81
	RZ	-0.30	0.74		RX	0.02	0.16
KneeLeft	RY	-0.01	1.47		RZ	-0.15	0.33

Table 5.3: Upper and lower bounds of the generalized positions $\boldsymbol{q}(t)$ for the gait synthesis problem. Axes with a T in the name denote a translational and with a R a rotational degree of freedom.

5.4.1 Problem Formulation

The optimal control problem of the gait synthesis is of the form of equation (5.1). Specifically this formulation uses the free parameter vector $\boldsymbol{p}$. It contains a parameters for the step length $p_l \in \mathbb{R}$ and walk velocity $p_v \in \mathbb{R}$ and also all the spring-damper parameters $\boldsymbol{p}_{sd}$ that we described in Section 4.5.

5.4.2 Objective Function

We have formulated multiple objective criteria that are used to formulate the composite objective function

$$\Phi(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} \left(\sum_{j=1}^{3} \gamma_j \Phi_{L_j}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) \right) dt + \gamma_4 \Phi_{M_4}(t_f, \boldsymbol{x}(t_f), \boldsymbol{p}). \tag{5.23}$$

The weighting factors $\gamma_j \in \mathbb{R}, j = 1, \ldots, 4$ can be adjusted to obtain different weightings of the individual objective criteria. The individual objective criteria are:

Minimize Active Actuation over Step Length

As a measure for the required effort to produce a step we use the objective function

$$\Phi_{L_1}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = \frac{1}{p_l n_u} || \boldsymbol{W} \boldsymbol{u}(t) ||_2^2, \tag{5.24}$$

which minimizes the weighted active actuation normalized by the step length p_l and the number of controls n_u.

The diagonal matrix $\boldsymbol{W} \in \mathbb{R}^{n_u \times n_u}$ contains weightings that normalize the different joint contributions and is the same matrix that is also used in the dynamics reconstruction problem Section 5.3. The same weightings are used for the active joint actuations on left and right side of the body.

Minimize Angular Momentum

Angular momentum describes the rotational momentum that the rigid-body system has about a certain point. In our case we use the angular momentum expressed at the center of mass. The objective function can be written as:

$$\Phi_{L_2}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = ||\boldsymbol{m}_{CM}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))||_2^2, \tag{5.25}$$

where $\boldsymbol{m}_{CM}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$ is said quantity. Its computation is presented in Section 3.4.5.

Minimize Head Angular Velocities

To formulate some kind of head stabilization criterion we formulated the objective function:

$$\Phi_{L_3}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = ||{}^0\boldsymbol{E}_{head}\ \boldsymbol{\omega}_{head}||_2^2. \tag{5.26}$$

Here ${}^0\boldsymbol{E}_{head}$ is the orientation of the head segment relative to the global reference frame its product with the angular velocity of the head $\boldsymbol{\omega}_{head}$ results in the angular velocity of the head segment in the global reference frame 0.

Minimize Step Time

The objective function that minimizes the duration of the step (not to be confused with the walk velocity) is stated as a Mayer term:

$$\Phi_{M_4}(t_f, \boldsymbol{x}(t_f), \boldsymbol{p}) = t_f. \tag{5.27}$$

It minimizes the time it takes to complete a single step. The Lagrange term is $\Phi_{L_4} = 0$.

5.4.3 Phase Descriptions

We use the same notation for the contact point descriptions as in the least-squares optimization formulation and also use the same phase names and order of phases. Furthermore, as we are interested in periodic and symmetric gaits one has to take care to prevent redundancies in the constraints that occur due to the symmetric periodicity.

Phase 1: *Right Flat*

The following constraints are posed at the beginning of the *Right Flat* phase:

$$\begin{aligned}
P_X^{Right,Hallux}(\boldsymbol{x}(t_0)) - P_X^{Left,Hallux}(\boldsymbol{x}(t_0)) - p_l &= 0,\\
F_Z^{Right,Heel}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0)) &\geq 0,\\
q_1(t_0) &= 0,\\
q_2(t_0) &= 0.
\end{aligned}$$

The first constraint ensures that the step is of length p_l. The next three constraints ensure that the contact forces are of proper sign, i.e. that the ground does not pull the model to the ground. The last two constraints ensure that the $X-$ and $Y-$ coordinates of the pelvis are at the beginning of the gait cycle. These constraints are crucial for the proper problem formulation. Without these there are infinitely many solutions for the periodic gait which are all shifted along these axes.

Additionally to the constraints above and the bounds in Table 5.3 we enforce a small amount of arm swinging that is anti-cyclic to the legs by setting

$$\begin{array}{rcccl} 0.03 & \leq & q_{ShoulderRight,RY}(t_0) & \leq & 0.89, \\ -0.51 & \leq & q_{ShoulderLeft,RY}(t_0) & \leq & 0.03. \end{array}$$

Within the phase we have:

$$\begin{aligned} F_Z^{Right,Heel}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1, \\ F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1, \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_1, \\ P_Z^{Left,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_1, \\ P_Z^{Left,Hallux}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_1, \\ P_Z^{Left,Meta5}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_1, \\ d(C_{RightShank}, C_{LeftShank}, \boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_1, \end{aligned}$$

again we have positive ground reaction forces on all contact points along the ground normal. Furthermore the left fore foot does not touch the ground and the boundary capsules of the left and right shank do not intersect.

Phase 2: *Right Forefoot*

At the beginning of the second phase the following constraints have to be fulfilled:

$$\begin{aligned} F_Z^{Right,Heel}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &= 0, \\ F_Z^{Right,Hallux}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0, \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0, \\ P_Z^{Left,Heel}(\boldsymbol{x}(t_2)) &\geq 0, \\ P_Z^{Left,Hallux}(\boldsymbol{x}(t_2)) &\geq 0, \\ P_Z^{Left,Meta5}(\boldsymbol{x}(t_2)) &\geq 0, \\ d(C_{RightShank}, C_{LeftShank}, \boldsymbol{x}(t_2)) &\geq 0. \end{aligned}$$

First, the constraint forces on the right heel must have vanished while the constraint force on the right Hallux point and the right Meta5 point is still positive along the ground normal. All three contact points of the left foot must not be below the ground.

During this phase the constraints are:

$$\begin{aligned} F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_2 \\ F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_2, \\ P_Z^{Left,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2, \\ P_Z^{Left,Hallux}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2, \\ P_Z^{Left,Meta5}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2, \\ P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2, \\ d(C_{RightShank}, C_{LeftShank}, \boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_2. \end{aligned}$$

The contact force on the right Hallux point must be positive and the contact points for the left foot and the right heel contact point must be nonnegative along the ground normal. Right and left shank must not intersect.

Transition 1: ***Collision Left Heel***

The first discontinuity occurs when only the right foot is in contact with the ground and the left comes into contact with the ground. This is described by the constraints

$$\begin{aligned}
P_Z^{Left,Heel}(\boldsymbol{x}(t_2)) &= 0,\\
V_Z^{Left,Heel}(\boldsymbol{x}(t_2)) &\leq 0,\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_2), \boldsymbol{u}(t_2)) &\geq 0,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t_2)) &\geq 0,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t_2)) &\geq 0,
\end{aligned}$$

The first two constraints are the same as in the least-squares optimization. Specifically the first equation ensures that the left heel is on ground level while the second confirms the touch-down. The next three equations ensure that the ground reaction forces on the forefoot are proper and the remaining that there is no intersection of the feet with the ground or the shank segments.

Again in this "phase" there is no integration. Instead the discontinuity in the generalized velocities $\dot{\boldsymbol{q}}$ due to the touch down collision is computed (see Section 3.1.3).

Phase 3: ***Right Forefoot Left Heel***

After the transition the phase *Right Forefoot Left Heel* starts. At the beginning and during this phase we require:

$$\begin{aligned}
F_Z^{Right,Hallux}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3,\\
F_Z^{Left,Heel}(\boldsymbol{x}(t), \boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_3,\\
P_Z^{Left,Hallux}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_3,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_3.
\end{aligned}$$

The left Hallux point and the right heel point should not be below the ground and the ground reaction forces on the two contact points right Hallux and left heel should be of proper sign.

Transition 2: ***Collision Left Forefoot***

The second discontinuity occurs when the left forefoot touches down. We formulate this using

$$\begin{aligned}
P_Z^{Left,Hallux}(\boldsymbol{x}(t_3)) &= 0,\\
P_Z^{Left,Meta5}(\boldsymbol{x}(t_3)) &= 0,\\
V_Z^{Left,Hallux}(\boldsymbol{x}(t_3)) &\leq 0,\\
V_Z^{Left,Meta5}(\boldsymbol{x}(t_3)) &\leq 0\\
F_Z^{Right,Hallux}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0,\\
F_Z^{Left,Heel}(\boldsymbol{x}(t_3), \boldsymbol{u}(t_3)) &\geq 0,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t_3)) &\geq 0,
\end{aligned}$$

The first four equation describe the touch-down of the forefoot as in the previous transition phase: the contact points are on ground level and have a negative contact velocity (i.e. a collision is imminent). The following three constraints ensure that the ground reaction forces of the current contact points are positive along the ground normal.

The last two constraints ensure that the right heel is not below the ground and the bounding capsules do not intersect.

Again, there is no integration in this phase and the discontinuities in the generalized velocities is updated such that the contact velocities are set to zero.

Phase 4: *Right Forefoot Left Flat*

At the beginning of this phase the left foot is flat on the ground and does not move due to the transition phase. In this final phase the weight is completely transferred onto the left foot. At the beginning and during this phase we have the constraints

$$\begin{aligned}
F_Z^{Right,Hallux}(\boldsymbol{x}(t),\boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t),\boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4,\\
F_Z^{Left,Heel}(\boldsymbol{x}(t),\boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4,\\
F_Z^{Left,Hallux}(\boldsymbol{x}(t),\boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4,\\
F_Z^{Left,Meta5}(\boldsymbol{x}(t),\boldsymbol{u}(t)) &\geq 0, t \in \mathcal{T}_4,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t)) &\geq 0, t \in \mathcal{T}_4.
\end{aligned}$$

Again, the contact forces must be of proper sign along the ground normal for the currently active contact points. Additionally the right heel must not be intersecting with the ground and the collision capsules must not intersect.

At the end of the phase we have a separate set of constraints that allow the right forefoot to be lifted from the ground:

$$\begin{aligned}
F_Z^{Right,Hallux}(\boldsymbol{x}(t_4),\boldsymbol{u}(t_4)) &= 0,\\
F_Z^{Right,Meta5}(\boldsymbol{x}(t_4),\boldsymbol{u}(t_4)) &= 0,\\
F_Z^{Left,Heel}(\boldsymbol{x}(t_4),\boldsymbol{u}(t_4)) &\geq 0,\\
F_Z^{Left,Hallux}(\boldsymbol{x}(t_4),\boldsymbol{u}(t_4)) &\geq 0,\\
F_Z^{Left,Meta5}(\boldsymbol{x}(t_4),\boldsymbol{u}(t_4)) &\geq 0,\\
P_Z^{Right,Heel}(\boldsymbol{x}(t_4)) &\geq 0.
\end{aligned}$$

The normal contact force of the right Hallux point and Meta5 point must vanish (first two equations), while the contact forces of the left foot are of proper sign along the Z-axis and the right heel is above the ground.

Periodicity and Symmetry

To formulate periodic symmetry we employ the symmetric state computation function $\boldsymbol{q}_{sym}$: $\mathbf{R}^{n_{dof}} \to \mathbf{R}^{n_{dof}}$. This function maps a given vector of generalized position variables $\boldsymbol{q}$ to a generalized position vector $\boldsymbol{q}^s$ that describes the original posture mirrored at the sagittal plane.

The same function can be used to compute the symmetric state on joint velocity level. We denote the symmetric generalized position of $\boldsymbol{q}$ by $\boldsymbol{q}^S$ and similarly the symmetric generalized velocity vector of $\dot{\boldsymbol{q}}$ by $\dot{\boldsymbol{q}}^S$.

The symmetric state computation function is given by:

$$\boldsymbol{q}_{sym}: \begin{bmatrix} q_{Pelvis,Translation,X} \\ q_{Pelvis,Translation,Y} \\ q_{Pelvis,Translation,Z} \\ q_{Pelvis,Rotation,Y} \\ q_{Pelvis,Rotation,X} \\ q_{Pelvis,Rotation,Z} \\ q_{Hip,Right,Y} \\ q_{Hip,Right,X} \\ q_{Hip,Right,Z} \\ q_{Knee,Right,Y} \\ q_{Ankle,Right,Y} \\ q_{Ankle,Right,X} \\ q_{Ankle,Right,Z} \\ q_{Hip,Left,Y} \\ q_{Hip,Left,X} \\ q_{Hip,Left,Z} \\ q_{Knee,Left,Y} \\ q_{Ankle,Left,Y} \\ q_{Ankle,Left,X} \\ q_{Ankle,Left,Z} \\ q_{MiddleTrunk,Right,Y} \\ q_{MiddleTrunk,Right,X} \\ q_{MiddleTrunk,Right,Z} \\ q_{Shoulder,Right,Y} \\ q_{Shoulder,Right,X} \\ q_{Shoulder,Right,Z} \\ q_{Elbow,Right,Y} \\ q_{Shoulder,Left,Y} \\ q_{Shoulder,Left,X} \\ q_{Shoulder,Left,Z} \\ q_{Elbow,Left,Y} \\ q_{Head,Y} \\ q_{Head,X} \\ q_{Head,Z} \end{bmatrix} \mapsto \begin{bmatrix} q_{Pelvis,Translation,X} \\ -q_{Pelvis,Translation,Y} \\ q_{Pelvis,Translation,Z} \\ q_{Pelvis,Rotation,Y} \\ -q_{Pelvis,Rotation,X} \\ -q_{Pelvis,Rotation,Z} \\ q_{Hip,Left,Y} \\ -q_{Hip,Left,X} \\ -q_{Hip,Left,Z} \\ q_{Knee,Left,Y} \\ q_{Ankle,Left,Y} \\ -q_{Ankle,Left,X} \\ -q_{Ankle,Left,Z} \\ q_{Hip,Right,Y} \\ -q_{Hip,Right,X} \\ -q_{Hip,Right,Z} \\ q_{Knee,Right,Y} \\ q_{Ankle,Right,Y} \\ -q_{Ankle,Right,X} \\ -q_{Ankle,Right,Z} \\ q_{MiddleTrunk,Right,Y} \\ -q_{MiddleTrunk,Right,X} \\ -q_{MiddleTrunk,Right,Z} \\ q_{Shoulder,Left,Y} \\ -q_{Shoulder,Left,X} \\ -q_{Shoulder,Left,Z} \\ q_{Elbow,Left,Y} \\ q_{Shoulder,Right,Y} \\ -q_{Shoulder,Right,X} \\ -q_{Shoulder,Right,Z} \\ q_{Elbow,Right,Y} \\ q_{Head,Y} \\ -q_{Head,X} \\ -q_{Head,Z} \end{bmatrix} \tag{5.28}$$

In addition to the periodicity of the joint positions and velocities we require periodicity of the joint actuations $\boldsymbol{u}(t)$. The symmetric control computation function $\boldsymbol{u}_{sym} : \mathbf{R}^{n_u} \to \mathbf{R}^{n_u}$ is given by

$$\boldsymbol{u}_{sym}: \begin{bmatrix} u_{Hip,Right,Y} \\ u_{Hip,Right,X} \\ u_{Hip,Right,Z} \\ u_{Knee,Right,Y} \\ u_{Ankle,Right,Y} \\ u_{Ankle,Right,X} \\ u_{Ankle,Right,Z} \\ u_{Hip,Left,Y} \\ u_{Hip,Left,X} \\ u_{Hip,Left,Z} \\ u_{Knee,Left,Y} \\ u_{Ankle,Left,Y} \\ u_{Ankle,Left,X} \\ u_{Ankle,Left,Z} \\ u_{MiddleTrunk,Right,Y} \\ u_{MiddleTrunk,Right,X} \\ u_{MiddleTrunk,Right,Z} \\ u_{Shoulder,Right,Y} \\ u_{Shoulder,Right,X} \\ u_{Shoulder,Right,Z} \\ u_{Elbow,Right,Y} \\ u_{Shoulder,Left,Y} \\ u_{Shoulder,Left,X} \\ u_{Shoulder,Left,Z} \\ u_{Elbow,Left,Y} \\ u_{Head,Y} \\ u_{Head,X} \\ u_{Head,Z} \end{bmatrix} \mapsto \begin{bmatrix} u_{Hip,Left,Y} \\ -u_{Hip,Left,X} \\ -u_{Hip,Left,Z} \\ u_{Knee,Left,Y} \\ u_{Ankle,Left,Y} \\ -u_{Ankle,Left,X} \\ -u_{Ankle,Left,Z} \\ u_{Hip,Right,Y} \\ -u_{Hip,Right,X} \\ -u_{Hip,Right,Z} \\ u_{Knee,Right,Y} \\ u_{Ankle,Right,Y} \\ -u_{Ankle,Right,X} \\ -u_{Ankle,Right,Z} \\ u_{MiddleTrunk,Right,Y} \\ -u_{MiddleTrunk,Right,X} \\ -u_{MiddleTrunk,Right,Z} \\ u_{Shoulder,Left,Y} \\ -u_{Shoulder,Left,X} \\ -u_{Shoulder,Left,Z} \\ u_{Elbow,Left,Y} \\ u_{Shoulder,Right,Y} \\ -u_{Shoulder,Right,X} \\ -u_{Shoulder,Right,Z} \\ u_{Elbow,Right,Y} \\ u_{Head,Y} \\ -u_{Head,X} \\ -u_{Head,Z} \end{bmatrix}. \tag{5.29}$$

We denote the symmetric controls of $\boldsymbol{u}(t)$ by $\boldsymbol{u}^S(t)$, similar to the symmetric state computation function. Periodic symmetry is then formulated as:

$$\boldsymbol{r}^{Sym}(t_0, t_f, \boldsymbol{x}(t_0), \boldsymbol{x}(t_f), \boldsymbol{u}(t_0), \boldsymbol{u}(t_f), \boldsymbol{p}) = \begin{bmatrix} \boldsymbol{r}^{Sym}_{pos}(t_0, t_f, \boldsymbol{q}(t_0), \boldsymbol{q}(t_f), \boldsymbol{p}) \\ \boldsymbol{r}^{Sym}_{vel}(\dot{\boldsymbol{q}}(t_0), \dot{\boldsymbol{q}}(t_f)) \\ \boldsymbol{r}^{Sym}_{act}(\boldsymbol{u}(t_0), \boldsymbol{u}(t_f)) \end{bmatrix} = \boldsymbol{0} \in \mathbb{R}^{2n_{dof}+n_u} \tag{5.30}$$

The n_{dof} boundary constraints $\boldsymbol{r}^{Sym}_{pos}(t_0, t_f, \boldsymbol{q}(t_0), \boldsymbol{q}(t_f), \boldsymbol{p})$ describe two characteristics of the gait: *i*) periodicity on the joint position level and *ii*) the average velocity along the walk direction. The latter is formulated in the first constraint as

$$q_1(t_0) - q_1(t_f) + (t_f - t_0)p_v = 0, \tag{5.31}$$

while the remaining $n_{dof} - 1$ constraints are simply positional symmetric periodicity constraints

$$q_i(t_0) - q_i^S(t_f) = 0. \tag{5.32}$$

The following n_{dof} boundary constraints are the periodicity conditions for the joint velocities:

$$\boldsymbol{r}^{Sym}_{vel}(\dot{\boldsymbol{q}}(t_0), \dot{\boldsymbol{q}}(t_f)) = \dot{\boldsymbol{q}}(t_0) - \dot{\boldsymbol{q}}^S(t_f) = \boldsymbol{0} \tag{5.33}$$

and the remaining n_u constraints are the periodicity conditions for the joint actuations:

$$\boldsymbol{r}^{Sym}_{act}(\boldsymbol{u}(t_0), \boldsymbol{u}(t_f)) = \boldsymbol{u}(t_0) - \boldsymbol{u}^S(t_f) = \boldsymbol{0}. \tag{5.34}$$

5.4.4 Initialization

As initial data for states, controls, step length, walk velocity, and phase durations we use results from the reconstruction problem described in Section 5.3. Since the results are not perfectly symmetric we empoloyed a manual homotopy method by requiring symmetric periodicity on only a subset of constraints and only performing a small number of SQP steps to reach sufficient feasibility. Further periodicity constraints were added until full periodicity was reached.

Once fully periodic initial values were available the SQP method was able to produce iterates even when setting the initial controls to $\boldsymbol{u}(t) = \mathbf{0}$ or initializing the differential states using linear interpolation of values at the phase boundaries. This re-initialization was also helpful to restart the optimization when the SQP got stuck in a local minimum.

Part III

Numerical Results and Outlook

Chapter 6

Numerical Results

In this chapter we present the numerical results of this thesis. We first demonstrate the effectiveness of the subject-specific modelling in terms of the kinematic fitting quality of the recorded motion data. We analyze the recorded kinematic data in terms of changes due to different emotional expressions. We show the reconstructed dynamics that we obtained using our subject-specific model and our multi-phase optimal control problem formulation and describe effects of changes in emotional expressions in the dynamic quantities. Further we show the synthesized full-body gaits that we computed solely based on mathematical modeling and numerical computation and compare them with the recorded emotional motions.

6.1 Subject-Specific Models

We created subject-specific models for Subject1 and Subject3 as their emotional expressions were best recognized. The model creation and adjustments were performed as described in Chapter 4.

6.1.1 Subject Parameters

Table 6.1 shows the adjusted parameters for Subject1 and Subject3 along with the deviations from a uniformly scaled model that linearly scales all lengths based on the subject height. Most length parameters are adjusted in the range of $3 - 5cm$ with respect to the uniformly scaled model. The largest deviation from the uniform model is for the *NeckLength* parameter at around $8cm$. The average difference of the length parameters between the two subjects is around $1.6cm$.

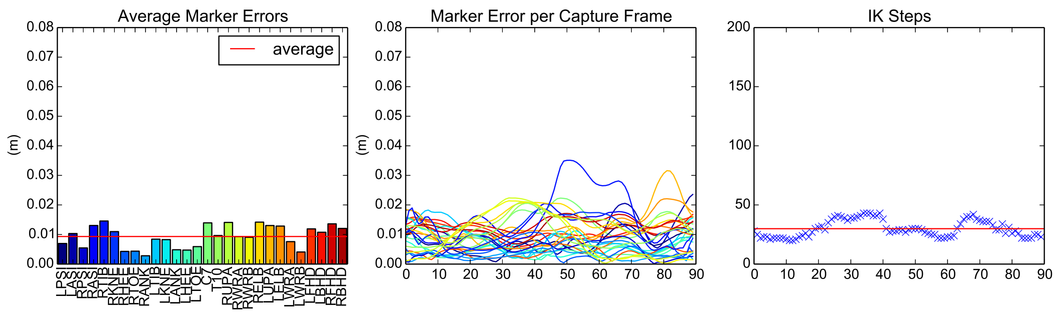
6.1.2 Kinematic Fit: Marker Data to Joint Positions

In Figure 6.1 we show how well the recorded motions can be reconstructed using the subject-specific models. To evaluate the effectiveness of the subject-specific modeling we performed two kinds of fittings: *i) congruent fitting* in which both the model and the recording were for the same subject and *ii) incongruent fitting* in which we used the model of one subject and the motion from another.

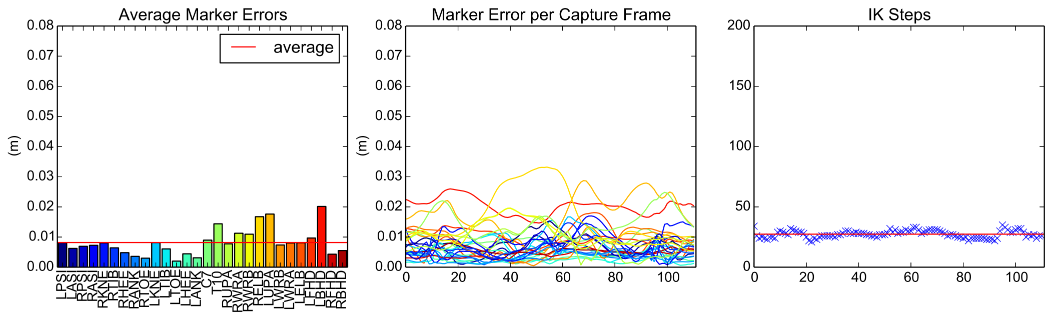
From the recorded motion capture data we chose trials of the expressions *AngerExpression*, which show the strongest articulations among all subjects and is therefore the most challenging to fit.

In Figure 6.1 we show the fitting analysis of the two congruent fittings and the two incongruent fittings. Each analysis consists of three plots: the average marker errors, marker errors for each motion capture frame, and the number of inverse kinematic steps.

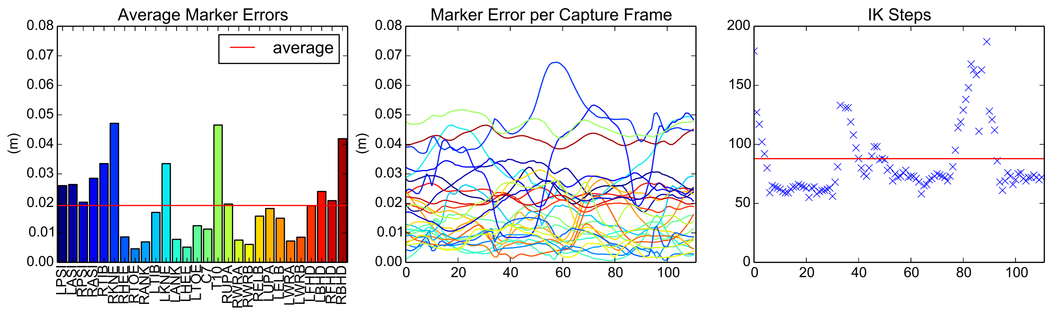
The average marker error plot shows the average error between the virtual marker on the model and the recorded marker over the whole motion sequence. It shows how well individual markers can reproduce the corresponding recorded marker. For a good model one would expect average errors of similar magnitude.



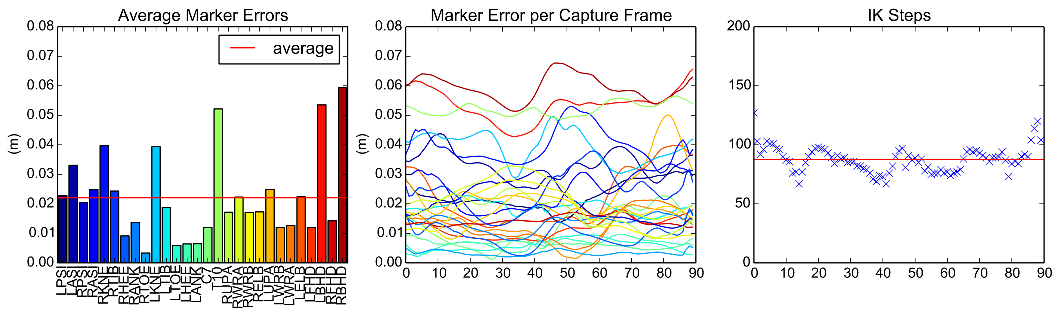
(a) Congruent fitting of *S1MAngerExpressive1.c3d* onto Subject1



(b) Congruent fitting of *S3MAngerExpressive2.c3d* onto Subject3



(c) Incongruent fitting of *S3MAngerExpressive2.c3d* onto Subject1



(d) Incongruent fitting of *S1MAngerExpressive1.c3d* onto Subject3

Figure 6.1: Fitting analysis of a *AngerExpressive* motion onto the subject specific models. For each subject we show the average residual for each marker. The overall average residual of all markers is shown on the left, the marker errors at each motion capture frame, and the number of inverse kinematics (IK) steps. For the congruent cases we use matching movement data and model for a specific subject. In the incongruent case we fit movement data of one subject onto the model of another subject.

Parameter	**Subject1** (deviation from default value)	**Subject3** (deviation from default value)
ClaviculaLeftLength	0.1799m (-0.0385m)	0.1749m (-0.0483m)
ClaviculaRightLength	0.1799m (-0.0385m)	0.1749m (-0.0483m)
FootLeftHeight	0.0852m (0.0000m)	0.0870m (0.0000m)
FootLeftLength	0.2661m (0.0000m)	0.2719m (0.0000m)
FootLeftPosteriorPoint	0.0728m (0.0000m)	0.0744m (0.0000m)
FootRightHeight	0.0852m (0.0000m)	0.0870m (0.0000m)
FootRightLength	0.2661m (0.0000m)	0.2719m (0.0000m)
FootRightPosteriorPoint	0.0728m (0.0000m)	0.0744m (-0.0000m)
HandLeftLength	0.1432m (-0.0532m)	0.1432m (-0.0575m)
HandRightLength	0.1432m (-0.0532m)	0.1432m (-0.0575m)
HeadLength	0.2497m (0.0372m)	0.2197m (0.0025m)
HipLeftWidth	0.0832m (-0.0007m)	0.0832m (-0.0025m)
HipRightWidth	0.0832m (-0.0007m)	0.0832m (-0.0025m)
LowerArmLeftLength	0.2764m (-0.0047m)	0.2714m (-0.0159m)
LowerArmRightLength	0.2764m (-0.0047m)	0.2714m (-0.0159m)
LowerTrunkHeight	0.1498m (-0.0025m)	0.1098m (-0.0459m)
MiddleTrunkHeight	0.1716m (-0.0537m)	0.2216m (-0.0087m)
NeckLength	0.0258m (-0.0902m)	0.0358m (-0.0828m)
ShankLeftLength	0.4827m (0.0224m)	0.4627m (-0.0077m)
ShankRightLength	0.4827m (0.0224m)	0.4627m (-0.0077m)
ShoulderLeftHeight	0.2239m (0.0455m)	0.2289m (0.0465m)
ShoulderRightHeight	0.2239m (0.0455m)	0.2289m (0.0465m)
SuprasternaleHeight	0.2239m (0.0455m)	0.2339m (0.0515m)
ThighLeftLength	0.4141m (-0.0273m)	0.4441m (-0.0070m)
ThighRightLength	0.4141m (-0.0273m)	0.4441m (-0.0070m)
UpperArmLeftLength	0.3596m (0.0651m)	0.3796m (0.0786m)
UpperArmRightLength	0.3596m (0.0651m)	0.3796m (0.0786m)
UpperTrunkHeight	0.2389m (0.0605m)	0.2339m (0.0515m)

Table 6.1: HeiMan parameters for segement lengths in meter for Subject1 and Subject3. In parenthesis we show the deviation from default value that are uniformly scaled by the subject height.

The second plot shows the error between virtual and recorded marker for each recorded frame. It gives insight on whether the error between model and recorded marker is dependent on the articulation of the recorded subject or rather due to noise from skin movement or measurement noise. If the error is low (e.g. $< 2cm$) except for a short range of frames (e.g. 20% of the frames) then this can be an indicator for secondary motion (e.g. skin movement) of the recorded marker.

Figure 6.1(a) and Figure 6.1(b) show the fitting analysis for the congruent cases. One can see an average error of approximately $1cm$ for both subjects and the error for individual markers is always less than $4cm$ for all markers in both subjects. The number of inverse kinematics steps is also relatively low with less than 30 steps for both subjects.

For the incongruent fittings shown in Figure 6.1(c) and Figure 6.1(d) one can see a still relatively low average error of around $2cm$, however the distribution of the average errors is widely spread. Individual markers have average errors of $4cm$ and more while others are well below $1cm$. Compared to the congruent case the variance of average errors is much higher here (congruent: $\pm 0.42cm$ and $\pm 0.38cm$, incongruent: $\pm 1.40cm$ and $\pm 1.39cm$).

Another observation is that the congruent fits need much fewer inverse kinematic iterations compared to the incongruent fittings, which were on average more than three times the amount of inverse kinematics steps.

Two conclusions may be drawn from this:

1. Considerably better kinematic fits are obtained using subject-specific models. With these we are able to reproduce the recorded data with average errors of less than $1cm$ between the virtual markers on the model and the recorded markers.

2. The subject-specific models successfully capture the walking kinematics of the recorded subject as indicated by a lack of systematic errors in the markers.

6.2 Kinematic Gait Analysis of the Recordings

For each recorded trial of Subject3 we computed the gait parameters *step length*, *step duration*, and *walk velocity*. The results are shown in Figure 6.2. The expressions of *AngerSupressive*, *Neutral*, and *JoySelfsufficient* show very similar values for all three gait parameters. *AngerExpressive* consistently has the highest step length and lowest step durations which also leads to highest walk velocity. The parameters for expressions of *JoyExpressive* are always between those of *AngerExpressive* and the remaining emotions.

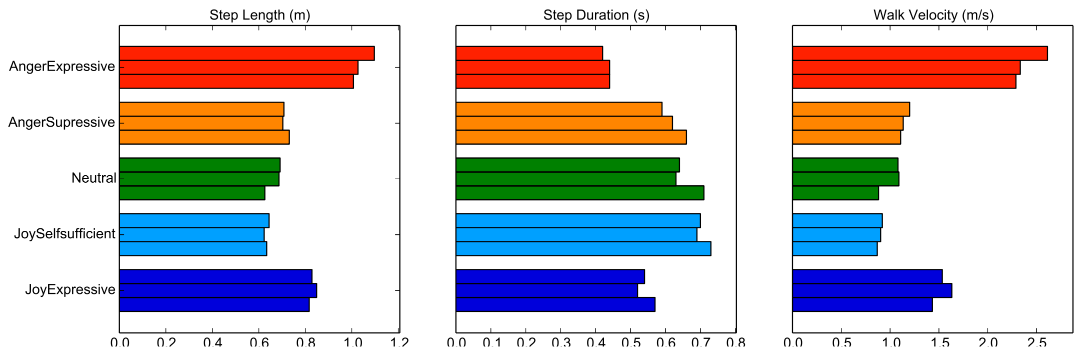


Figure 6.2: Gait parameters for the trials of Subject3. We show step lengths (left), step durations (middle), and walk velocities (right) for the recorded trials.

6.2.1 Patterns in the Joint Kinematics

In Figure 6.3 we show the trajectories of the generalized positions that we computed by fitting the motion capture marker data onto the subject-specific model as described in Section 4.4.3. For each plot we show the average of each emotion as a solid line and the standard deviation as a lightly shaded area. Plots of the individual trials are shown in Appendix A.2.

In the plot of the pelvis translation along the sagittal axis one can see large differences of the lines of *AngerExpressive* in red at the top, followed by the blue line for *JoyExpressive*. The remaining expressions *Neutral, AngerSupressive*, and *JoySupressive* are close together, which simply restates the characteristics of the step lengths also shown in Figure 6.2. Effects of increase in the step lengths can also be seen in the plots `Pelvis_RZ`, `HipRight_RY`, `HipLeft_RY`, `KneeRight_RY`, and `KneeLeft_RY`, where the largest range of angles are observed for *AngerExpressive*, followed by *JoyExpressive* and *AngerSupressive*.

The expression of *AngerExpressive* can be discriminated in multiple plots. E.g. in plot `Pelvis_TZ`, which shows translation along the longitudonal axis, this expression has the lowest values throughout the step. This is likely due to the larger step length, which has to be compensated by a lower pelvis due to the constant leg length. The plot `Pelvis_RY` shows a higher value for the rotation of the pelvis around the transverse axis. Compared to the other expressions this results in a slightly forward bent pelvis. Similar observations can be made for the plot of `Lumbar_RY` for which *AngerExpressive* and *AngerSupressive* both show the highest values.

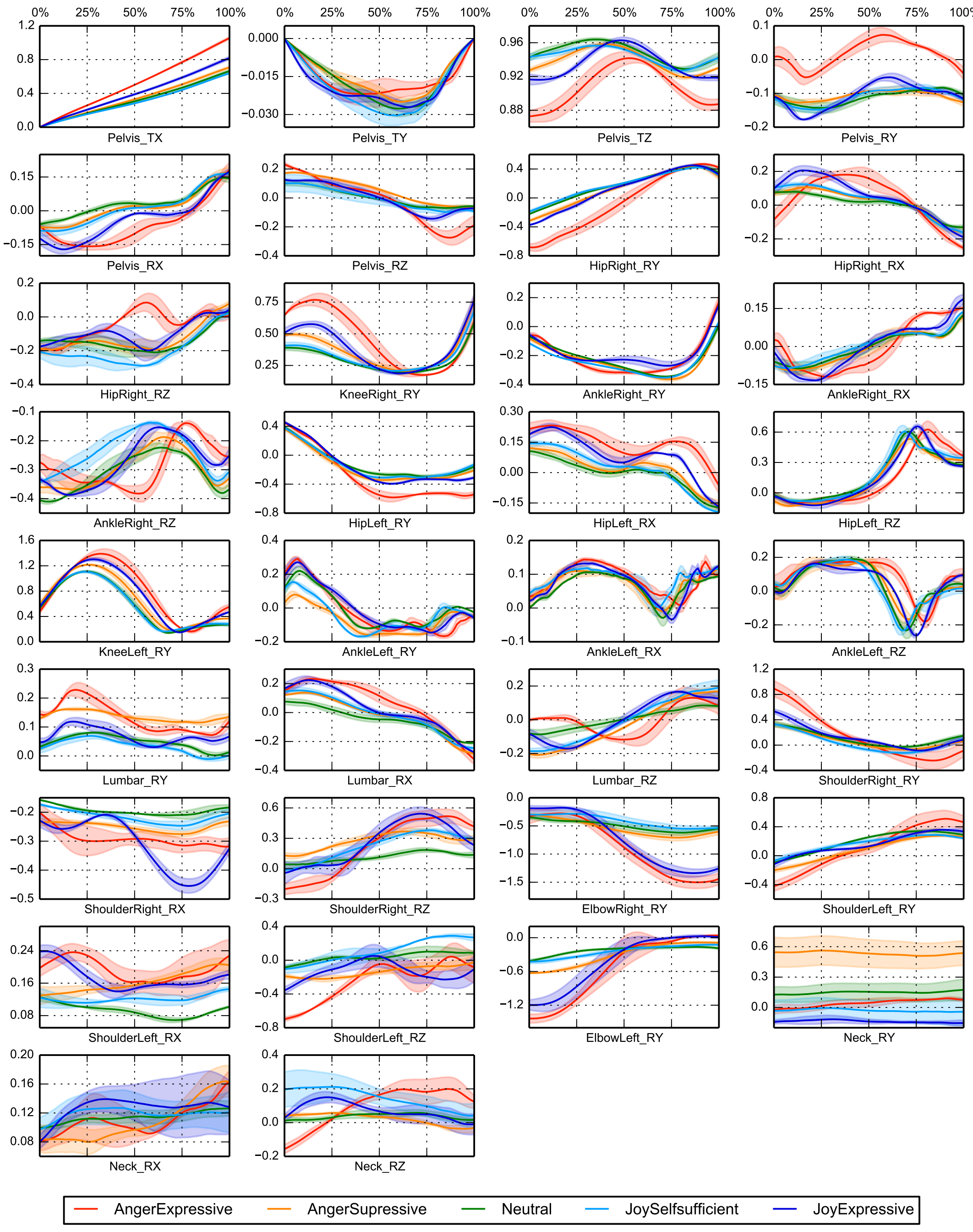


Figure 6.3: Generalized positions of the emotional expressions *AngerExpressive, AngerSupressive, Neutral, JoySelfsufficient*, and *JoyExpressive* that we computed from the motion capture marker data for the subject-specific model of Subject3. We show the average as a line and standard deviation as an area, based on three expressions of Subject3 for each emotion. Plots ending with `_TX`,`_TY`,`_TZ` show translations in meters along the X–, Y–, and Z–axis and those ending with `_RX`,`_RY`, and `_RZ` show rotations in radians around the corresponding axes. Plots of the individual trials are shown in Appendix A.2.

The expressive motions *JoyExpressive* and *AngerExpressive* both feature similar trajectories and a much larger range of values in the plots `ElbowRight_RY` and `ElbowLeft_RY` compared to the other expressions. This supports the characterizations of Wallbott (1998), where both *Joy* and *Anger* are described as "expansive" and "high movement dynamic". Another interesting observation can be made for the abduction of the right shoulder shown in plot `ShoulderRight_RX`: here *JoyExpressive* has the largest range of values and also the overall shape of the trajectory is different compared to the other expressions.

Expressions of *AngerSupressive* and *JoySupressive* are difficult to discriminate from other expressions. Only the plot of `Neck_RY`, which corresponds to the flexion of the neck, shows a consistently high value for the expression of *AngerSupressive*.

6.3 Reconstruction of the Dynamics using Least-Squares Fitting

We reconstructed the gait dynamics for three expressions each of the emotions *AngerExpressive, AngerSupressive, Neutral, JoySelfsufficient*, and *JoyExpressive*. For this we have used the recordings of Subject3, as they were well recognized in our expressiveness, whose results we presented in Section 2.3.2.

For each recording we analyzed the motion capture data to find contact events such as *Heel Touchdown*, *Heel Liftoff*, *Forefoot Touchdown*, and *Forefoot Liftoff*. These events were then used to extract motion sequences and phase timings that correspond to the phases of our dynamics reconstruction optimal control problem formulation described in Section 5.3. We then used our subject-specific model of Subject3 to perform a kinematic fit of the joint angles for the motion sequences. For this we used our modeling tool PUPPETEER, which we described in Section 4.4. We use the resulting joint angles both as an initial guess for the optimal solution and also in the objective function to minimize the joint residuals, i.e. error between joint angles of the model and those of from the kinematic fit.

In Table 6.2 we show the multiple-shooting discretization settings used for the different trials and dimensions of the resulting discretized nonlinear programming problem. One should note for the nshoot column that the third and fifth phases are pseudo-phases that model the discontinuities of contact collisions, i.e. no integration is performed for these. As base functions for the control discretization we use piecewise linear functions.

Table 6.3 lists the weighting coefficients that we used for the regularization term of the least-squares objective function as described in Section 5.3.2. We computed the $\boldsymbol{w}_i^{-1}$ by solving the optimal control problem (5.21) for the *Neutral* motions using $\gamma_u = 0$ and computing the average generalized joint force for each degree of freedom. We further used the same weighting coefficients on the left and right side of the model by averaging their values.

For each of the five emotional expressions we reconstructed the dynamics of three motion capture trials, which gives us a total of 15 optimal control problems that we implemented and solved.

6.3.1 Fit Analysis

To evaluate how well the dynamic subject-specific model approximates the kinematic joint angles we compared the trajectories of the generalized positions of the reconstruction with the joint angles from the kinematic fit.

In Figure 6.4 we show the average and standard deviation of the errors for each degree of freedom over all trials. Generally the errors are larger for values that are affected by the contact constraints and very small average errors for joints belonging to the upper body. The absolute value of the errors is within $\pm 0.01\ rad \approx \pm 0.57°$ for the rotational degrees of freedom and below $1cm$ for translational degrees of freedom except for the vertical movement of the pelvis which has an average error of slightly above $2cm$.

In Figure 6.5 we show a visual representation of the difference between the motion from the dynamic LSQ fit and the reference motion capture data for expressions of *AngerExpressive*,

Trial	nshoot	NDIS	NVAR	NEQ	NINEQ
AngerExpressive2	(6, 12, 1, 3, 1, 1)	25	3072	2376	6258
AngerExpressive3	(8, 10, 1, 3, 1, 1)	25	3072	2376	6254
AngerExpressive4	(8, 12, 1, 3, 1, 1)	27	3320	2568	6760
AngerSupressive1	(7, 11, 1, 4, 1, 4)	29	3568	2760	7271
AngerSupressive2	(9, 8, 1, 3, 1, 3)	26	3196	2472	6507
AngerSupressive3	(6, 10, 1, 4, 1, 3)	25	3196	2472	6513
Neutral1	(6, 13, 1, 3, 1, 5)	30	3692	2856	7527
Neutral2	(6, 12, 1, 3, 1, 3)	27	3320	2568	6766
Neutral3	(7, 10, 1, 5, 1, 2)	27	3320	2568	6763
AngerSelfsufficient1	(9, 10, 1, 3, 1, 5)	30	3692	2856	7521
AngerSelfsufficient2	(5, 13, 1, 4, 1, 4)	29	3568	2760	7275
AngerSelfsufficient3	(6, 12, 1, 4, 1, 5)	30	3692	2856	7527
AngerExpressive1	(7, 13, 1, 4, 1, 2)	29	3568	2760	7269
AngerExpressive2	(5, 13, 1, 3, 1, 2)	26	3196	2472	6514
AngerExpressive3	(6, 12, 1, 3, 1, 2)	26	3196	2472	6512

Table 6.2: Discretization settings and discretized NLP problem dimensions. For each trial the nshoot column contains the number of multiple- shooting intervals for each phase, NDIS are the number of discretization points, NVAR the number of variables in the discretized NLP, NEQ the number of equality constraints, and NINEQ the number of inequalities.

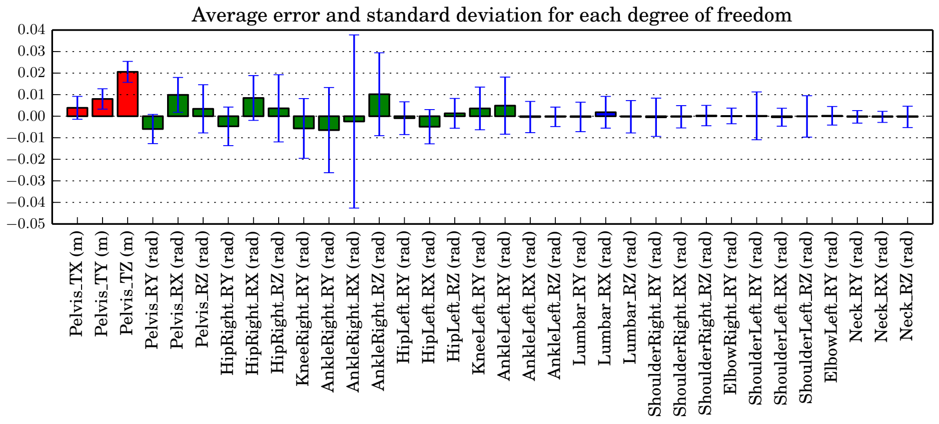


Figure 6.4: Average errors and standard deviation of the fitted residuals. Red bars represent the translation of the pelvis in meter, green bars the rotational degrees of freedom for the lower body, and the blue bars those of the upper body.

Neutral, and *JoyExpressive*. Similar plots for *AngerSupressive* and *JoySelfsufficient* are shown in Appendix A.1.

The dynamic fit motion is shown in the foreground in green while the reference motion is shown in the background in red. They are shown using orthographic projection such that the model in the background has the same size as the model in the front to allow a comparison of the two.

The difference between the motions is barely visible and only for a few postures one can see clear differences. One of them is the last image of the lateral view of *AngerExpressive* where only about 2/3 of the hand area overlap.

For the *AngerExpressive* and *JoyExpressive* motions one can also notice a slight error in the hip abduction. However one should note that the motion of the dynamic fit not only

Joint	**Motion**	**Axis**	w_i^{-1}
Hip	Flexion/Extension	Y	15.5
Hip	Adduction/Abduction	X	18.0
Hip	Internal/Medial Rotation	Z	3.6
Knee	Flexion/Extension	Y	14.4
Ankle	Flexion/Extension	Y	39.4
Ankle	Adduction/Abduction	X	5.2
Ankle	Internal/Medial Rotation	Z	2.5
Lumbar	Flexion/Extension	Y	5.6
Lumbar	Adduction/Abduction	X	13.9
Lumbar	Internal/Medial Rotation	Z	2.5
Shoulder	Flexion/Extension	Y	1.5
Shoulder	Adduction/Abduction	X	1.8
Shoulder	Internal/Medial Rotation	Z	0.2
Elbow	Flexion/Extension	Y	0.8
Neck	Flexion/Extension	Y	0.8
Neck	Adduction/Abduction	X	1.0
Neck	Internal/Medial Rotation	Z	0.1

Table 6.3: Weighting coefficients to account for the different joint strengths.

approximates the reference data but is also subject to kinematic constraints at the feet. Even with these constraints our dynamic model can very well reproduce the given motions.

6.3.2 Elevation Angles

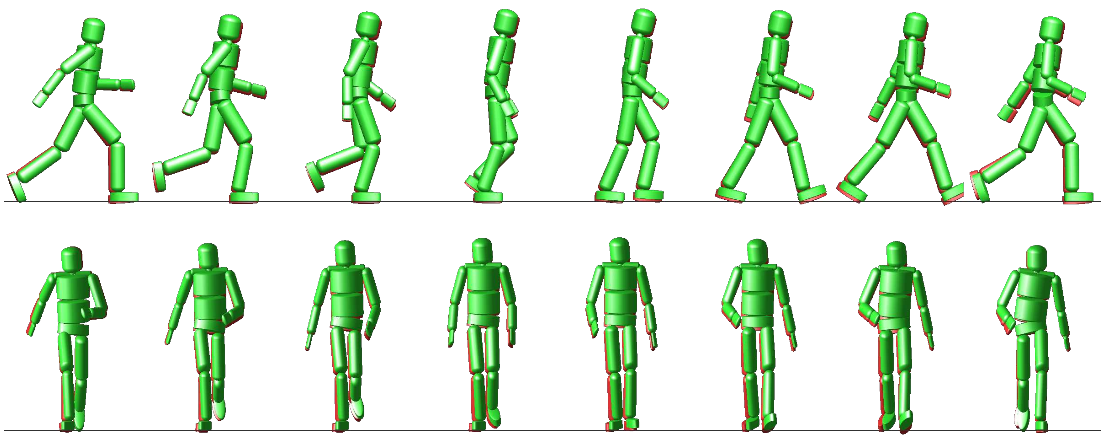
In Figure 6.7 we visualize the so-called *elevation angles* for both the kinematic fitting of the motion capture data (Figure 6.7(a)) and the kinematics of the model after the dynamic fitting (Figure 6.7(b)). The elevation angles are the orientations of the thigh, shank, and ankle segments measured in the sagittal plane of the global reference frame. Various studies have shown a linear relationship between these three angles (Borghese et al., 1996; Hicheur et al., 2006; Ivanenko et al., 2008) also for different tasks such as running or uphill stepping. The linear relationship of the elevation angles can be seen by plotting the angles onto each other in a 3-D plot and when looking from a specific angle one can see that the 3-D trajectories are close to a plane.

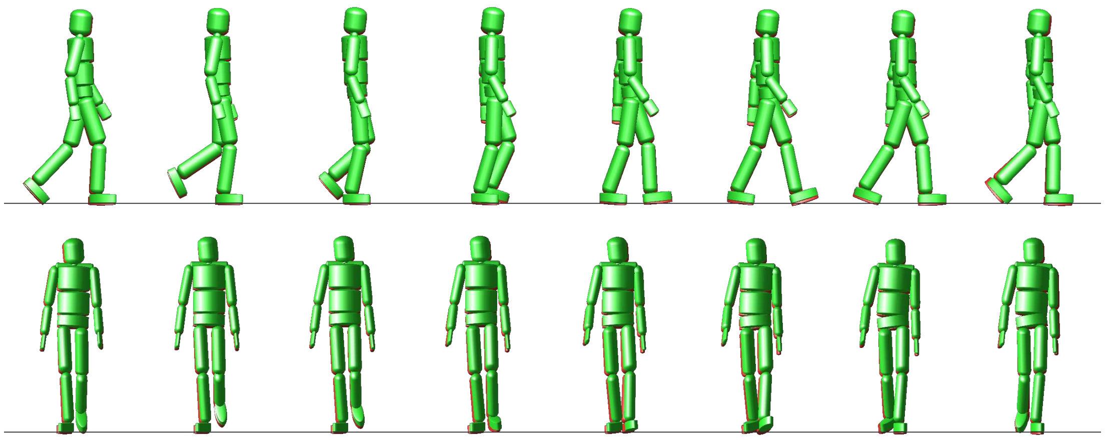
The angles we visualize here are converted to degrees. The left plots of Figure 6.7 show the overall shape from a diagonal perspective whereas the right plot is from a axis aligned perspective that shows the planarity of the data.

For each emotional expression we visualize the angles of a single step and the angles for the left and right leg are shown as two separate lines, which results in two open sections of the curves. The open sections can be seen in near the bottom left and the top right of the diagonal perspective plots.

When comparing the plots of the elevation angles of the kinematic fitting and the dynamic reconstruction one can see that the overall structure is very well preserved. However, when looking closely one can see slight jittering for the angles of the dynamic reconstruction in the perspective view, especially in the top right. Another difference can be seen in the axis aligned plots: near the 20° foot angle the profiles of the dynamic reconstruction show an almost horizontal plateau and a sharp corner, whereas the kinematic fitting is much smoother and no corners are visible. This is due to the rigid contact modeling of the dynamic reconstruction problem as the foot's orientation is kept perfectly horizontal when it is flat on the ground, which is reflected in a horizontal line in the elevation angles plot. In the kinematic fit we do not have a contact constraint and the foot angle is free to change as the foot undergoes small deformations due to the body weight.

In terms of changes due to emotions one can observe that the expressive emotions *An-*


(a) *AngerExpressive*


(b) *Neutral*

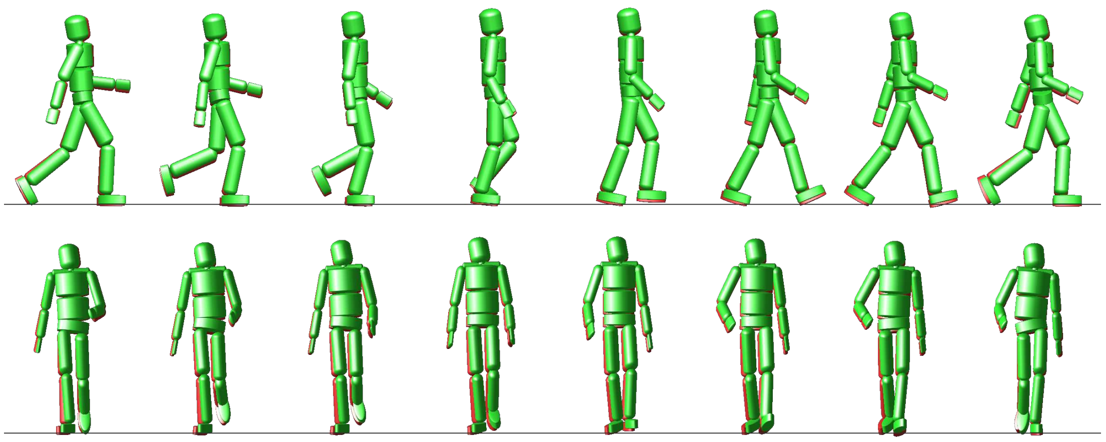

(c) *JoyExpressive*

Figure 6.5: Visualization of the movements and difference between the movement of the dynamic LSQ fit (shown in green) and reference movement from motion capture (shown in red). Each movement is shown in lateral and frontal view. Our dynamic model can to closely reproduce the reference motion even with the additional contact constraints.

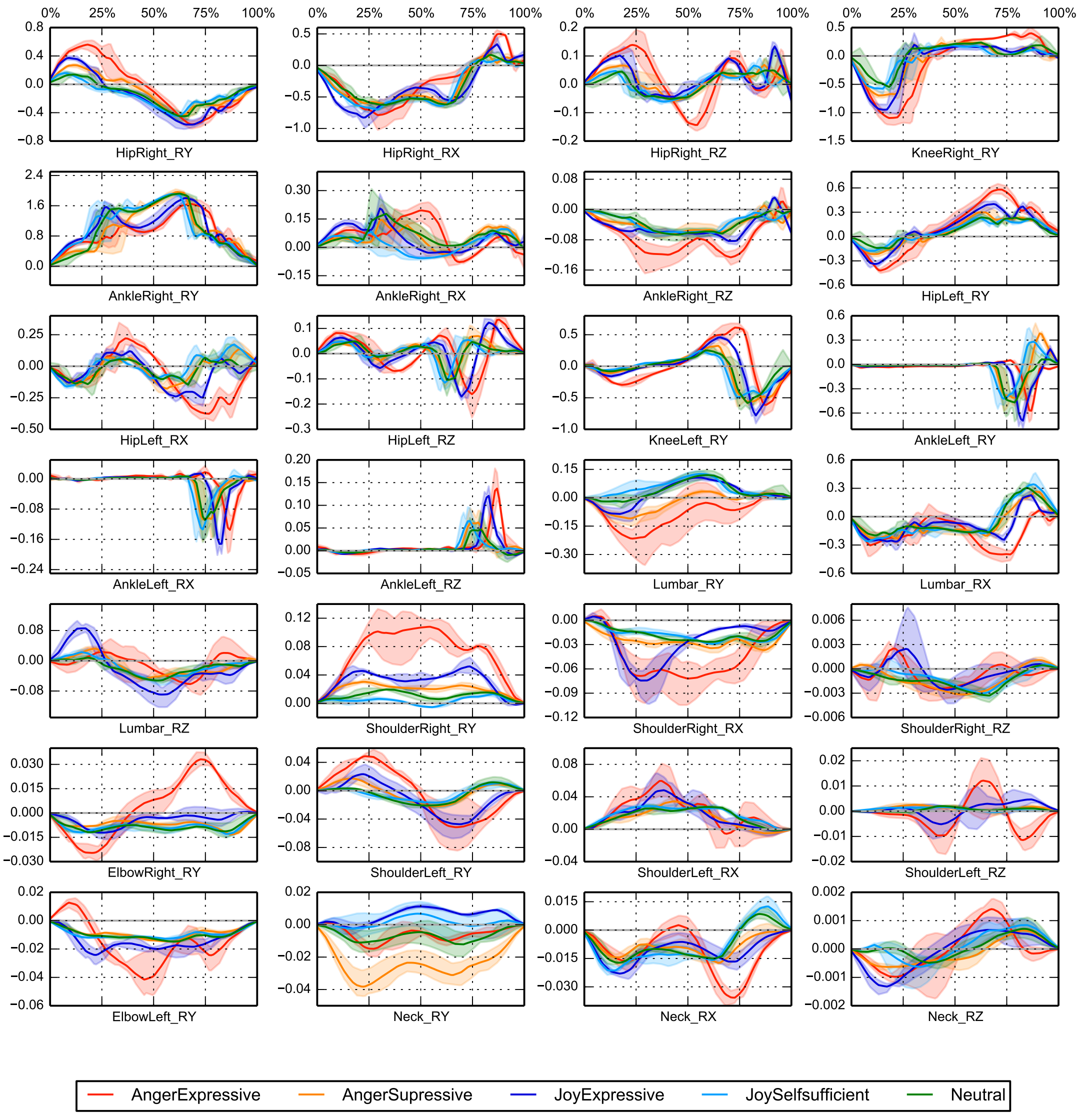
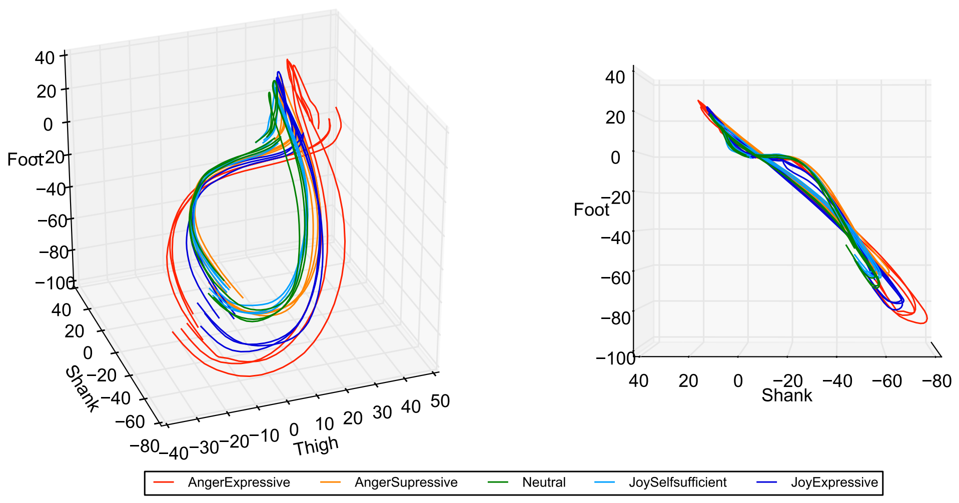
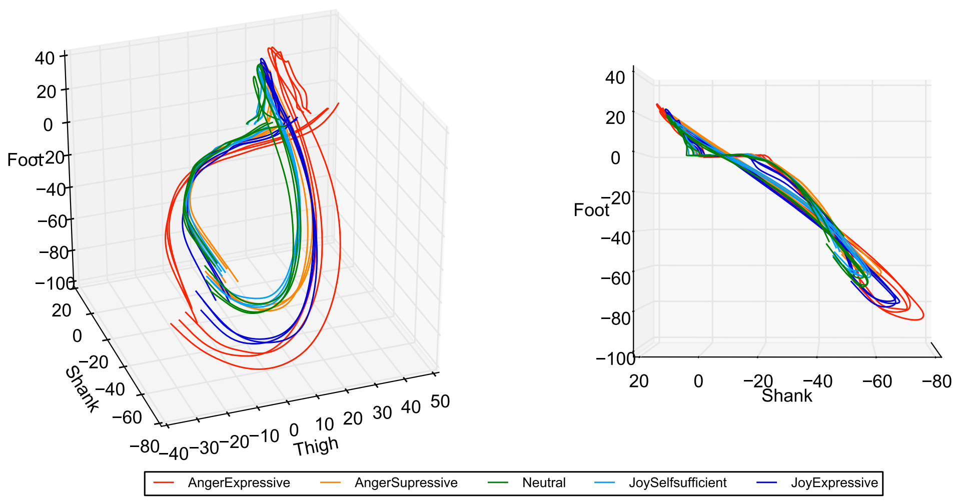


Figure 6.6: Reconstructed normalized joint torques for the emotional expressions *AngerExpressive, AngerSupressive, Neutral, JoySelfsufficient*, and *JoyExpressive*. The torques are normalized by the subject's body weight and for each subplot and emotional expression. We show the average as a line and the standard deviation as an area. Plots of the individual trials are shown in Appendix A.2.

gerExpressive and *JoyExpressive* result in larger spatial expansion of the curves compared to *AngerSupressive*, *Neutral* and *JoySelfsufficient* motions. This is likely due to the different walking velocities of the expressions as similar observations were made for changes in the walking velocity.



(a) Angles of the kinematic fitting of the motion capture data.



(b) Angles of the dynamic reconstruction.

Figure 6.7: Elevation angles for the different emotional expressions shown from a diagonal perspective (left) and axis aligned perspective (right).

6.3.3 Patterns in the Joint Torques

The plots of Figure 6.6 show the reconstructed joint torques of the emotional expressions. Again the movements of *AngerExpressive* generally show the largest amplitudes followed by *JoyExpressive*. Especially for the torqes of joints that perform flexion and extension, i.e. `Lumbar_RY`, `ShoulderRight_RY`, `ShoulderLeft_RY`, `ElbowRight_RY`, and `ShoulderLeft_RY` one can observe high levels of actuations for *AngerExpressive* compared to the other emotions.

In the plot `ShoulderRight_RX` the expression of *JoyExpressive* shows a different trajectory profile compared to *AngerExpressive*. This suggests that the shoulder abduction is not due to passive motion and but is instead actively controlled. Also the joint torques of `Lumbar_RZ`, which is the torque at the Lumbar joint around the longitudinal axis, shows large amplitudes for *JoyExpressive* and temporally shifted maxima and minima compared to *AngerExpressive*. Also the actuation of the shoulder flexion and extension shown in `ShoulderRight_RY` and `ShoulderLeft_RY` show the second largest amplitudes for *JoyExpressive*.

The expression of *AngerSupressive* clearly shows the largest amplitudes for the forces of the neck extension shown in the plot `Neck_RY`. Already the joint angle for this degree of freedom showed the highest values and could indicate that the control of this movement plays an important role in concealed anger.

We only found few indicators for patterns that are specific for *JoySelfsufficient*. In general the joint torque patterns are very similar to those of *Neutral*. In the first 25% of the step in plot of `Lumbar_RY` one can see slightly higher values for *JoySelfsufficient* compared to the other expressions. Also the actuation of in `Neck_RY` shows higher values of the joint torques for the joyful motions compared to the neutral and angry expressions.

6.3.4 Shared Patterns in EMG and Joint Torques

Comparing the EMG signals of Figure 2.7 and joint torques in Figure 6.6 can only be made on a qualitative basis. The relation of EMG signal and actual muscle force is in general very complex and depends on multiple factors such as muscle length, tension, velocity of shortening or lengthening of the muscle, fatigue, and others. EMG recordings in general only capture a subset of muscle fibers of a specific muscle, however the force for a specific movement may be caused by complex interactions with other muscles and other tissues. Computing the actual force based on EMG signals is still an active area of research (e.g. Sartori et al. (2012)). Nevertheless high EMG activity correlates with increased muscle force, which motivates the following comparisons. One should note that all EMG sensors were placed on the right side of the recorded subject and therefore we focus our comparison with joints on the right side of the subject-specific model.

The EMG signals for the expressions of *AngerExpressive* are consistently the highest for all recordings. A similar observation can be made for the joint torques where *AngerExpressive* shows the largest amplitudes for most joints.

One of the associated functions of the muscle *Gastrocnemius Lateralis* is the flexion of the knee. When looking at the EMG signal one can see that apart from *AngerExpressive* the muscular activity of the expressions *AngerSupressive*, *JoySelfsufficient*, and *JoyExpressive* are similar or below that of *Neutral*. In our model the flexion of the knee is described by positive values of `KneeRight_RY`. Similarly to the EMG signal the joint torque plots of this movement show *AngerExpressive* as largest values near the end of the right contact foot, whereas the remaining expressions are on a similar magnitude below the *AngerExpressive* joint torque.

Another characteristic can be seen by comparing the EMG activity of *Peroneus Longus* and the joint torques of `AnkleRight_RX`. The muscle contributes to the plantar flexion of the foot and the expression of *AngerExpressive* shows the largest values along with a temporal delay compared to the other expressions. A similar observation can be made when examining the joint torques of `AnkleRight_RX`. Here *AngerExpressive* has a delayed peak near the 50% mark.

The muscle *Vastus Medialis* contributes to the extension of the knee, which corresponds to a negative value of `KneeRight_RY`. Both *AngerExpressive* and *JoyExpressive* show high EMG activity during the early contact phase, wheras the expressions of *AngerSupressive*, *Neutral*, and *JoySupressive* are on a similar, albeit lower level of activity. The same observation can be made in the first quarter of the joint torques of `KneeRight_RY` where the expressive emotions both reach $-1.0Nm/kg$, whereas the remaining are at around $-0.5Nm/kg$.

6.3.5 Ground Reaction Forces

By reconstructing the dynamics we also obtain ground reaction forces that the model exerts on the ground. In Figure 6.8 we show the ground reaction forces for the different emotional expressions. The plots in the top row show the forces for the right foot along the $X-$, $Y-$, and $Z-$ axis, whereas the bottom plots show the forces for the left leg along the same axes.

Overall the force profiles are similar to that of reported measurements in the biomechanics literature such as Lee and Hidler (2008), especially those of the $X-$ and $Z-$ axis of the right leg. The forces of the left leg show the forces starting from heel contact to flat contact. They start with non-zero values as the initial contact is treated as an instantaneous collision in our walking model. One would expect that the forces of the left leg at the end of the step are similar to those of the right leg at the beginning of the step. In our results however this is not the case. This artefact is caused by our choice of gait modeling: as we only optimize a single step instead of a double-step sequence the optimization omits parts of the preparation for the second step.

The forces of the right leg show the for human walking characteristic M-shape for the vertical forces (Z-axis). The different emotions exhibit changes in the profile similar to those observed for different walking speeds (Masani et al., 2002). Specifically the less pronounced valley of the M at slower speeds (*AngerSupressive, Neutral, JoySelfsufficient* and the higher first bump for higher walking speeds (*AngerExpressive, JoyExpressive*).

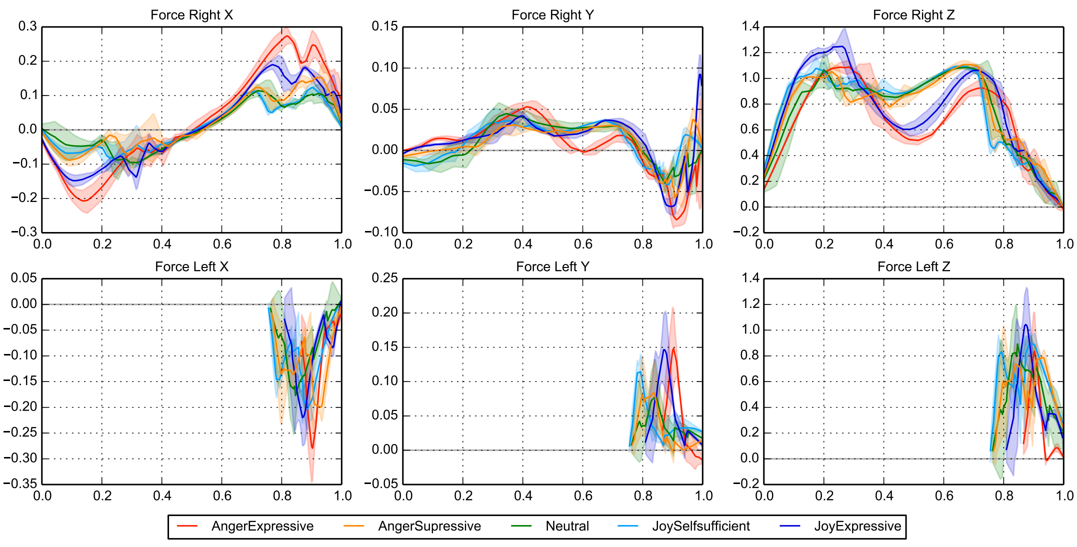


Figure 6.8: Reconstructed ground reaction forces of the different emotional expressions. The plots are time and weight normalized. Solid lines show the average for a single emotion and the standard deviation is shown as a shaded area.

6.4 Gait Synthesis

In this section we describe the computed walking motions that we generated with our gait synthesis formulation described in Section 5.4. By formulating different objective functions and gait parameters we are able to generate distinct walking motions based solely on mathematical principles and modeling.

6.4.1 Description of the Objective Functions

In Section 5.4.2 we presented the composite objective function (5.23), which we reproduce here for clarity:

$$\Phi(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) = \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} \left(\sum_{j=1}^{3} \gamma_j \Phi_{L_j}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}) \right) dt + \gamma_4 \Phi_{M_4}(t_f, \boldsymbol{x}(t_f), \boldsymbol{p}),$$

with the basic objective functions:

- $\Phi_{L_1}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p})$, which minimizes the active actuation over step length,

- $\Phi_{L_2}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p})$, which minimizes the centroidal angular momentum,
- $\Phi_{L_3}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p})$, which minimizes the head angular velocity,
- and $\Phi_{M_4}(t_f, \boldsymbol{x}(t_f), \boldsymbol{p})$, which minimizes the step duration.

The weightings $\gamma_1, \dots, \gamma_4$ can then be chosen freely to formulate different objective functions.

We used the following weightings to construct five different objective functions:

	Name	γ_1	γ_2	γ_3	γ_4
Φ_1	*Minimize Actuation over Step Length*	1.0	0.0	0.0	0.0
Φ_2	*Minimize Angular Momentum*	0.01	1.0	0.0	0.0
Φ_3	*Minimize Head Angular Velocity*	0.01	0.0	10.0	0.0
Φ_4	*Minimize Step Duration*	0.01	0.0	0.0	10.0
Φ_5	*Multi Objective*	1.0	1.0	3.0	0.0

The objective function Φ_1 (*Minimize Actuation over Step Length*) only minimizes the active actuation over the step length. This objective function can be interpreted as the amount of effort to perform the step. The objective function Φ_2 (*Minimize Angular Momentum*) primarily minimizes the angular momentum. It also minimizes the active actuation with a small factor of $\gamma_1 = 0.01$ to regularize the problem. Without the regularization the controls may take arbitrary values without affecting the actual solution. The criterion Φ_3 (*Minimize Head Angular Velocity*) minimizes the head angular velocity together with the active actuation. The weightings are chosen to account for the smaller magnitude of the angular velocity of the head ($\gamma_3 = 10.0$) and to regularize the problem ($\gamma_1 = 0.01$). By mainly minimizing the head angular velocity this can be seen as a head stabilization criteria. The cost function Φ_4 (*Minimize Step Duration*) minimizes the duration of the step. Again it combines two base functions. The largest contribution is from the Φ_{M_4} base objective function with a weighting of $\gamma_4 = 10.0$ and the second term again to regularize the problem ($\gamma_1 = 0.01$). Finally the objective function Φ_5 (*Multi Objective*) minimizes with respect to multiple basic objective criteria, namely the active actuation over step length ($\gamma_1 = 1.0$), angular momentum ($\gamma_2 = 1.0$), and the head angular velocity ($\gamma_3 = 3.0$).

For the direct multiple-shooting discretization we used $(10, 2, 1, 2, 1, 2)$ multiple-shooting intervals for each phase. Again the third and fourth phases are pseudo-phases where the contact collision discontinuities are treated. The controls are discretized using piecewise linear base functions. After the discretization the problem has 2381 variables and 1887 equality and 4885 inequality constraints.

Visualizations

In Figure 6.9 and Figure 6.10 we show visualizations of the gaits that we synthesized using the different objective functions from frontal and lateral view, respectively. The trajectories of the generalized joint positions, the generalized joint forces and the contributions of the spring-dampers, and plots of the spring-damper parameters can be found in Appendix A.3.

Only the gaits produced by *Minimize Actuation over Step Length* and *Multi Objective* extend the knee during stance such that it is a straight line. By doing so the model can reduce the torques applied on the knee. The same motions show a significant amount of elbow flexion. The upper-body posture is almost completely upright throughout the gait cycle compared to the other motions which appear to be slightly bent forward.

From the frontal view in Figure 6.10 one can see further differences of the gaits. By looking for each motion at the fourth image one can see different postures for each gait. *Minimize Actuation over Step Length* shows a slightly outward rotated left leg and the highest foot lifting, *Minimize Angular Momentum* has its left leg closely adducted to the right leg, for *Minimize Head Angular Velocity* the overall posture looks like a relaxed standing pose, while *Minimize Step Duration* looks similar however with the feet much closer to each other. The motions of *Minimize Actuation over Step Length*, *Minimize Angular Momentum*, and *Multi Objective* show an asymmetric posture whereas *Minimize Head Angular Velocity* and *Minimize Step Duration* are nearly symmetric.

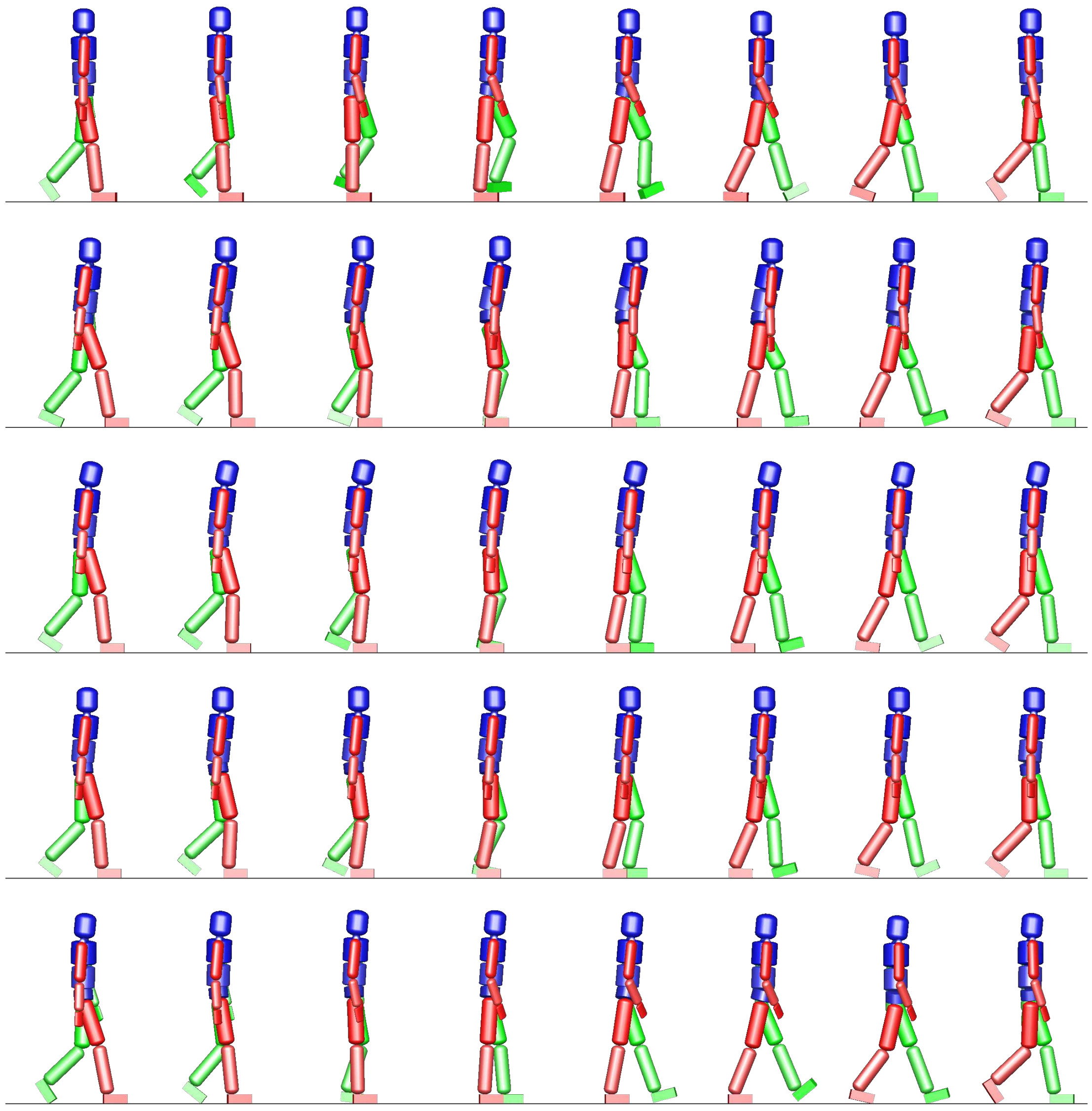


Figure 6.9: Lateral view of the synthesized open-loop gaits for the objective functions *Minimize Actuation over Step Length*, *Minimize Angular Momentum*, *Minimize Head Angular Velocity*, *Minimize Step Duration*, and *Multi Objective*.

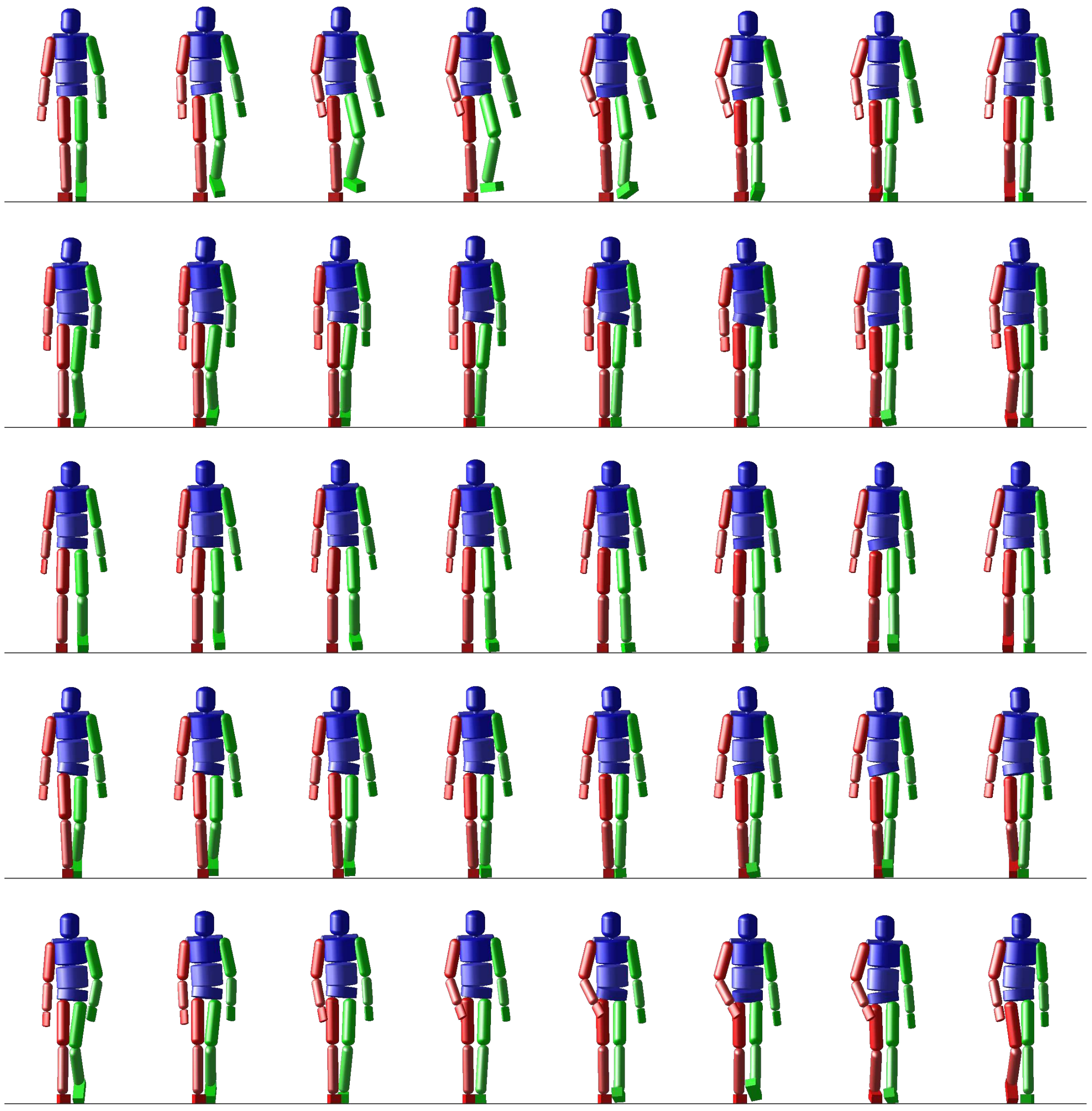


Figure 6.10: Frontal view of the synthesized open-loop gaits for the objective functions *Minimize Actuation over Step Length*, *Minimize Angular Momentum*, *Minimize Head Angular Velocity*, *Minimize Step Duration*, and *Multi Objective*.

6.4.2 Variation of Gait Parameters

In addition to the gaits synthesized by using different objective functions we have also generated gaits by prescribing the average walk velocity. In the gait synthesis formulation the average walk velocity is formulated as the parameter p_v, which is a free parameter by default. Here we have fixed its value to $1.5m/s, 2.0m/s$, and $2.5m/s$ to generate gaits for these velocities. We used the *Multi Objective* objective function as it resulted in motions that were the most similar to the *Neutral* motion capture motions.

We used the same discretization settings as for the variation of objective function formulation above. As the average walk velocity parameter is fixed the dimensions of the discretized optimal control problem change to: 2380 free variables, 1887 equality constraints, and 4883 inequality constraints.

Visualizations

In and Figure 6.11(a) and Figure 6.11(b) we show visualizations of gaits that we generated for the walk velocity variations. The trajectories of the generalized joint positions, the generalized joint forces and the contributions of the spring-dampers, and plots of the spring-damper parameters can be found in Appendix A.4.

We observe a much stronger arm movement for the slowest motion. Conversely, on higher speeds we observe a strong lateral pelvis tilt that is much stronger than for the $1.5m/s$ motion which can be seen in the fourth images of the frontal views in Figure 6.11(b). For the slowest motion the knee is extended during stance such that thigh and shank form a line. In the higher speeds the knee is always slightly flexed throughout the step.

6.4.3 Comparison of Synthesized and Reconstructed Emotional Motions

To compare the synthesized gaits with the reconstructed gaits we formulated the following similarity measure to compare two motions given as trajectories of generalized joint positions $\boldsymbol{q}^S(t)$ and $\boldsymbol{q}^M(t)$ with $\boldsymbol{q}^S(t) : [0, t_f^S] \to \mathbb{R}^{n_{dof}}$ and $\boldsymbol{q}^M(t) : [0, t_f^M] \to \mathbb{R}^{n_{dof}}$:

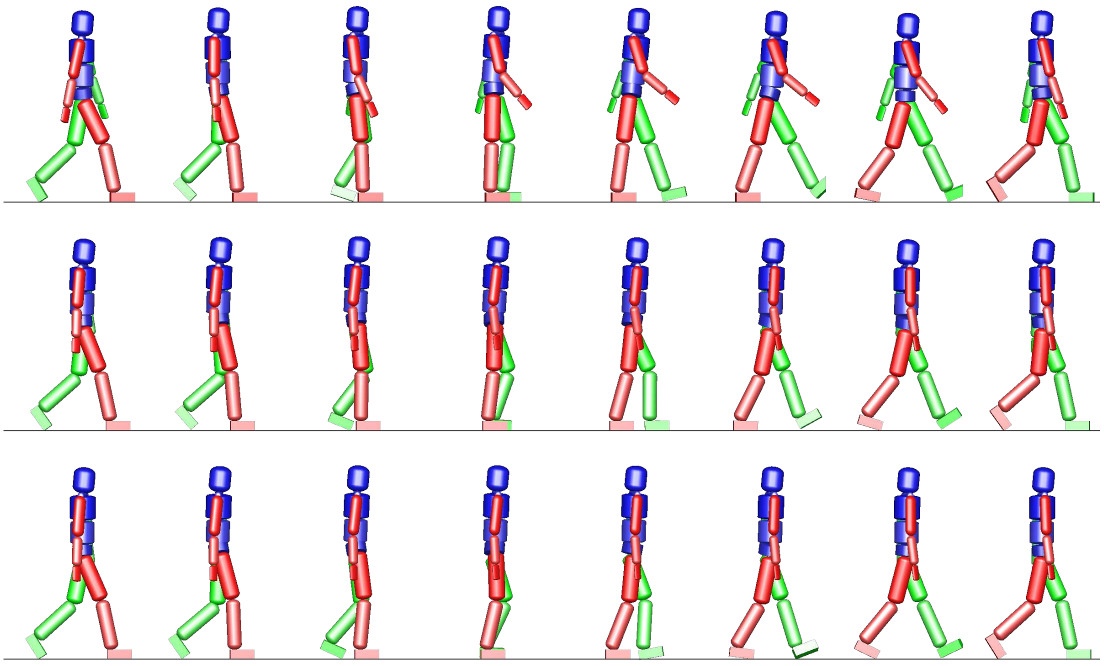
$$d(\boldsymbol{q}^S(t), \boldsymbol{q}^M(t), t_f^S, t_f^M) = \frac{1}{2}\sum_{i=1}^{m}\left(\frac{1}{m \cdot n_{dof}}||\boldsymbol{q}_i^S(i\Delta t^S) - \boldsymbol{q}_i^M(i\Delta t^M)||_2^2\right) + \frac{1}{2}(t_f^S - t_f^M)^2, \quad (6.1)$$

with $\Delta t^S = t_f^S/(m-1)$ and $\Delta t^M = t_f^M/(m-1)$. The similarity measure evaluates the difference of the two motions in terms of a least-square error of the generalized joint positions on a equidistant time grid and the difference of the step durations. The error of the generalized joint positions is normalize both by the number of evaluations m and the number of degrees of freedom n_{dof}.

To evaluate the similarity of the synthesized gaits with the measured movements we computed a *Neutral* reference motion by averaging the time-normalized joint trajectories $\boldsymbol{q}(t)$ of all three *Neutral* expressions. Similarly we computed the averaged *Neutral* step duration that we use as the reference *Neutral* duration.

Similarity Emotional Expressions and *Neutral*

We evaluated the similarity of the recorded emotional motions with the *Neutral* reference motion. The results are shown in Table 6.4 where we list for each expression the average similarity value and its standard deviation. As expected the *Neutral* motions have the smallest values with an average value of 0.0010 ± 0.0001 and the largest value is obtained for the *AngerExpressive* motions. With an average value of 0.0054 ± 0.0008 the *JoySelfsufficient* expressions are the closest to the *Neutral* reference motions. We note here that the similarity values of these two expressions also explains the similar ratings for the evaluation of the expressiveness of the recordings that we described and presented in Section 2.3.2.


(a) Lateral view synthesized gaits for different walking velocities.

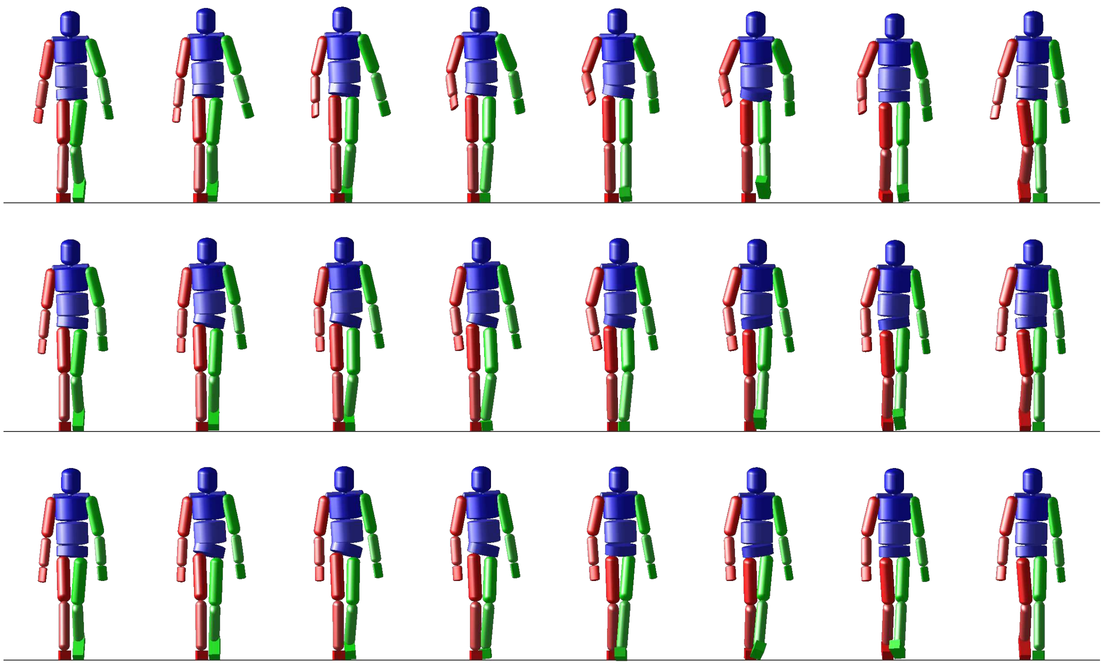

(b) Frontal view synthesized gaits for different walking velocities.

Figure 6.11: Visualization of synthesized gaits for different walking velocities (each from top to down): $1.5m/s, 2.0m/s, 2.5m/s$.

Solution of	$d(\boldsymbol{q}^S, \boldsymbol{q}^M, t_f^S, t_f^M)$	LSQ $\boldsymbol{q}$	LSQ t_f
AngerExpressive	0.0463±0.0016	0.0459±0.0016	0.0515±0.0043
AngerSupressive	0.0110±0.0035	0.0106±0.0035	0.0022±0.0020
Neutral	0.0010±0.0001	0.0006±0.0001	0.0013±0.0009
JoySelfsufficient	0.0054±0.0008	0.0050±0.0008	0.0025±0.0017
JoyExpressive	0.0241±0.0020	0.0237±0.0020	0.0140±0.0047

Table 6.4: Evaluation of the similarity of the recorded gaits and the *Neutral* reference motion. For each emotional expression we show the average value and standard deviation for each the similarity value (second column), the contribution of the difference in the postures (LSQ $\boldsymbol{q}$), and the contribution of the difference in the end time (LSQ t_f).

Similarity of the Synthesized Motions and *Neutral*

Here we compare the similarity of the *Neutral* reference motion with synthesized gaits obtained by variation of the objective function. The results are shown in Table 6.5 for an evaluation of $m = 100$. The table shows both the value of the similarity measures and the individual contributions of the terms dependent on $\boldsymbol{q}$ and step durations t_f.

The objective function *Multi Objective* has best similarity value, followed by *Minimize Actuation over Step Length* and *Minimize Angular Momentum*. The worst is *Minimize Step Duration*, which is however due to the large contribution of the t_f term. The solution of *Minimize Head Angular Velocity* shows the smallest error in the least-squares term of the joint angles. A visual comparison of the solution using *Multi Objective* and a *Neutral* expression is given Figure 6.12.

Solution of	$d(\boldsymbol{q}^S, \boldsymbol{q}^M, t_f^S, t_f^M)$	LSQ $\boldsymbol{q}$	LSQ t_f
Minimize Actuation over Step Length	0.1027	0.0718	0.0309
Minimize Angular Momentum	0.1455	0.0723	0.0732
Minimize Head Angular Velocity	0.1935	0.0374	0.1562
Minimize Step Duration	0.2761	0.0645	0.2116
Multi Objective	0.0699	0.0695	0.0244

Table 6.5: Evaluation of the similarity of the synthesized gaits and the *Neutral* reference motions. The similarity value is shown in the second column. LSQ $\boldsymbol{q}$ shows the contribution of the difference in the postures and LSQ t_f shows the contribution of the difference in the end time.

6.4.4 Elevation Angles

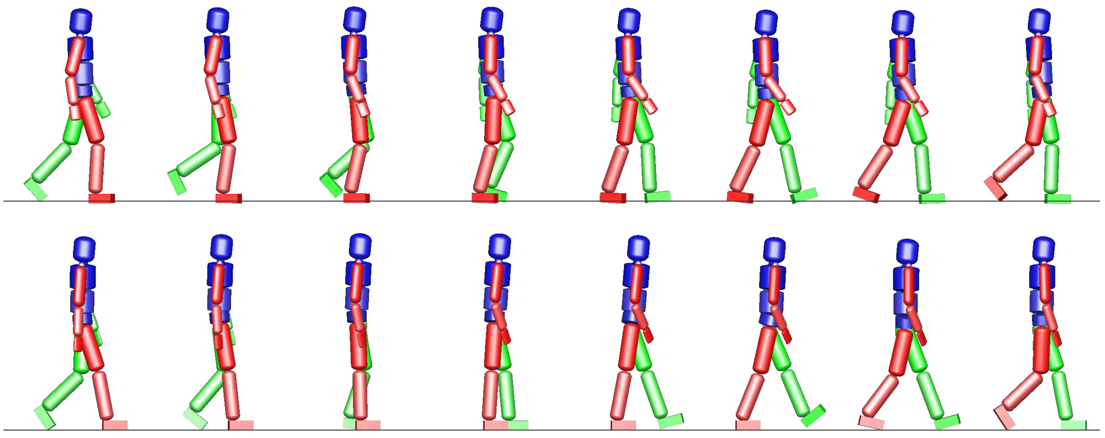
To assess the quality of the lower limb kinematics we computed the elevation angles for the gaits obtained using our synthesis formulation and compared it with the curves of the reconstructed gaits.

Objective Function Variations

In Figure 6.13(a) we show visualizations of the elevation angles for the synthesized gaits as colored lines along with the curves of the reconstructed gaits shown using grey lines.

The curves of the synthesized gaits are smoothly closed as the movements are perfectly symmetric. The overall alignment of the curves of the synthesized gaits is similar to that of the reconstructed gaits, however a few differences can be observed:

The flat plateau for the foot contact at the 0° value of the foot elevation angle is larger than for the reconstructed gaits. Outside of this plateau the motion of *Minimize Actuation over Step Length* shows a almost vertical feature near the −50° value of the Shank in the right figure,


(a) Lateral view of motion capture (top) and synthesized (bottom)

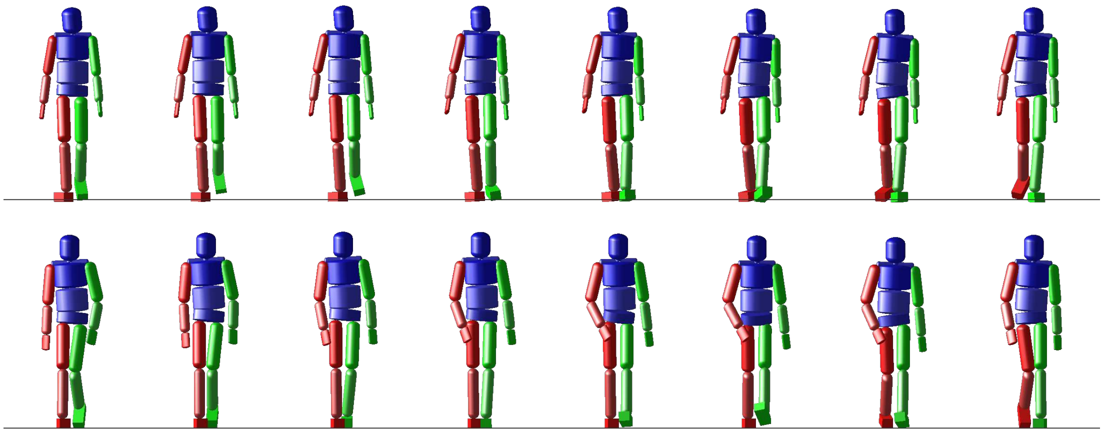

(b) Frontal view of motion capture (top) and synthesized (bottom)

Figure 6.12: Comparison of motion capture data (top) with the synthesized gait using the objective function *Multi Objective*.

while the other objective function evaluations are closer to the lines of the reconstructed gaits. Further, the elevation angles of the synthesized gaits have values in a smaller range than the reconstructed gaits.

Walking Velocity Variations

In Figure 6.13(b) we show the elevation angles for the synthesized gaits for the prescribed average walking velocities.

The shape of the curves are similar to the curves of the different objective functions. Again the synthesized gaits have elevation angles on smaller range for the thigh and shank elevation angle. The curves of the synthesized gaits are almost vertical again near the $-50°$ shank elevation angle.

Comparing the elevation angles of both synthesis formulations with those of the reconstructed gaits shows that the curves of the synthesized gaits are much less planar than those of the reconstructed gaits. As the planarity is observable for the reconstructed gaits we do not assume it to be a limitation of the kinematic abilities of the underlying rigid multibody or the foot contact model. Instead we assume that this is due to our actuation model which does not contain modeling of intersegmental passive elements that span more than one limb.

During the gait reconstruction the objective function rewards the more natural planarity of the curves that is already present in the reference motion as shown in Figure 6.7(a). During the gait synthesis nothing results coupling of the segments, instead all degrees of freedom are completely independent of each other.

6.4.5 Ground Reaction Forces

We show the ground reaction forces of the synthesized gaits in Figure 6.14 as colored lines together with the reconstructed ground reaction forces of the emotional gaits shown grey areas as reference. The forces are normalized with respect to body weight.

Objective Function Variations

The top right plot shows the vertical ground reaction force of the right foot. Except for *Minimize Actuation over Step Length* we observe strong peaks at around 60% of the step cycle that exceed the maximum values of the reconstructed forces by far.

The characteristic M-shape of the vertical ground reaction force is difficult to recognize but can be seen when looking at the forces of both right and left leg. The first peak at the left foot at around 80% of the step cycle and then proceeds to the lowest level near 25% of the step cycle of the right foot and the second peak at 60% of the step cycle of the right foot.

Walking Velocity Variations

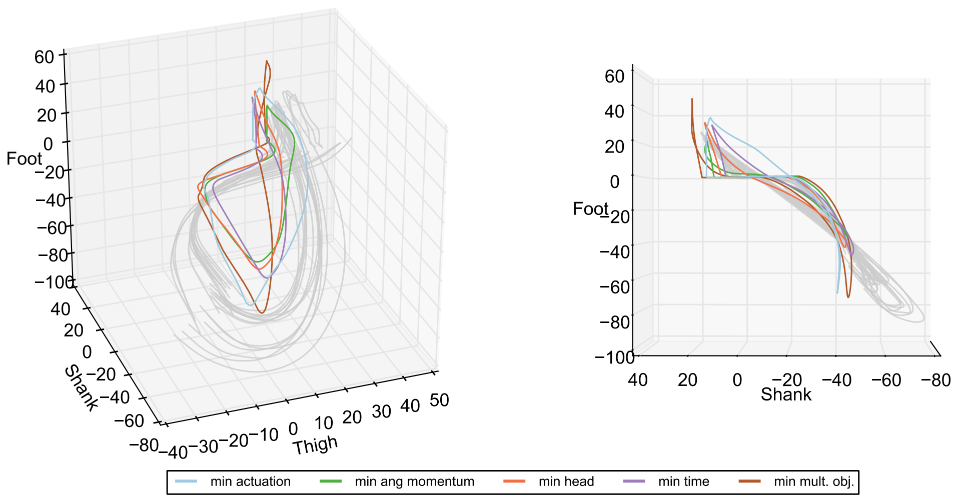
In Figure 6.14(b) we show the ground reaction forces for the synthesized gaits of different walking velocities.

The orders of magnitude of the forces are similar to those of the reconstructed gaits. Also the overall shape of the forces are similar, except for the forces along the Y-axes. Here the forces of the right foot do not switch sign and the forces of the left foot are of opposite sign than the reconstructed forces.

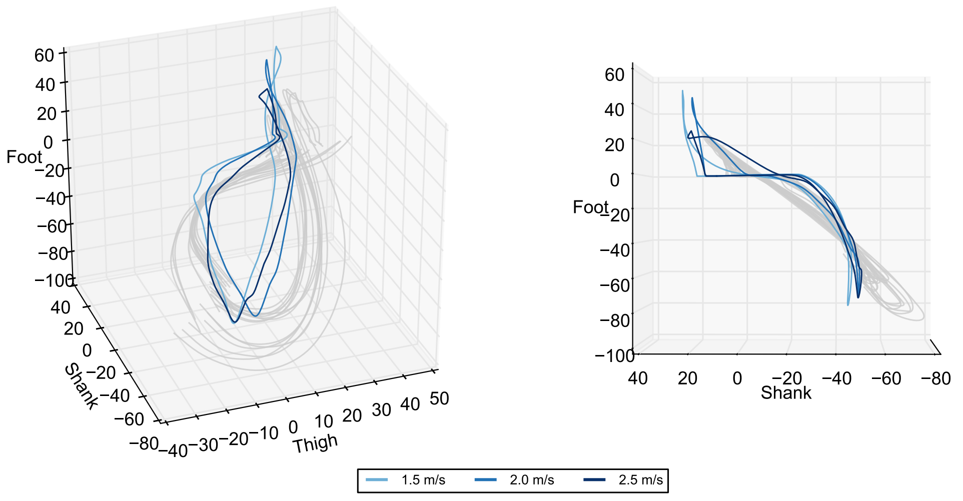
6.4.6 Objective Function Contributions

As shown above a different choice of weightings for the basic objective functions produces different gaits. As our long-term goal is to investigate the optimality criteria for different emotions there are two questions that arise from this observation: *i*) how much do the objective functions influence each other?, and *ii*) how much do the different basic objective functions contribute to emotional expressions?

To shed some light into these questions we use the solutions obtained (both from the gait reconstruction and gait synthesis) and evaluated the basic objective functions.

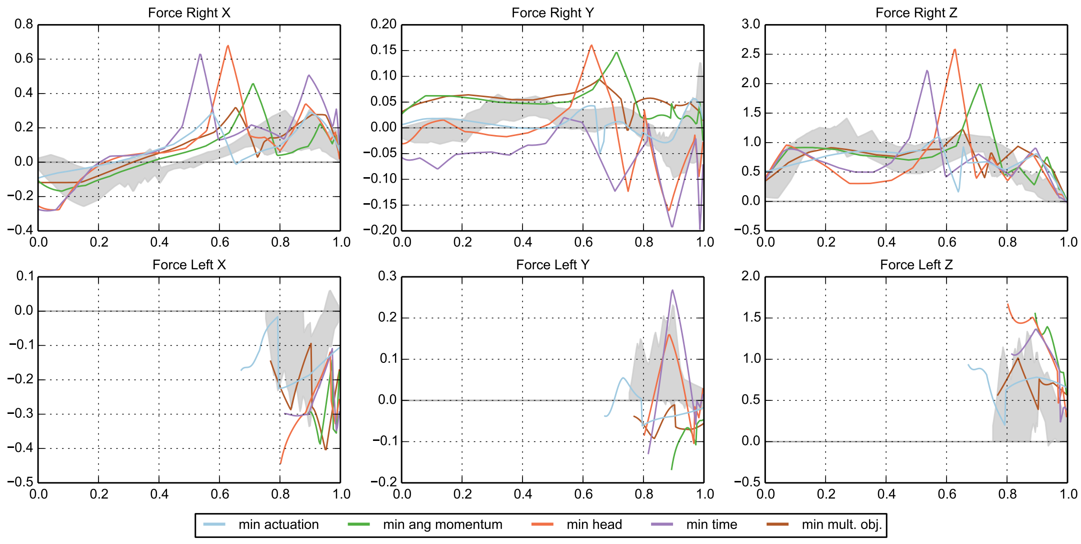


(a) Elevation angles of the synthesized gaits for different objective functions.

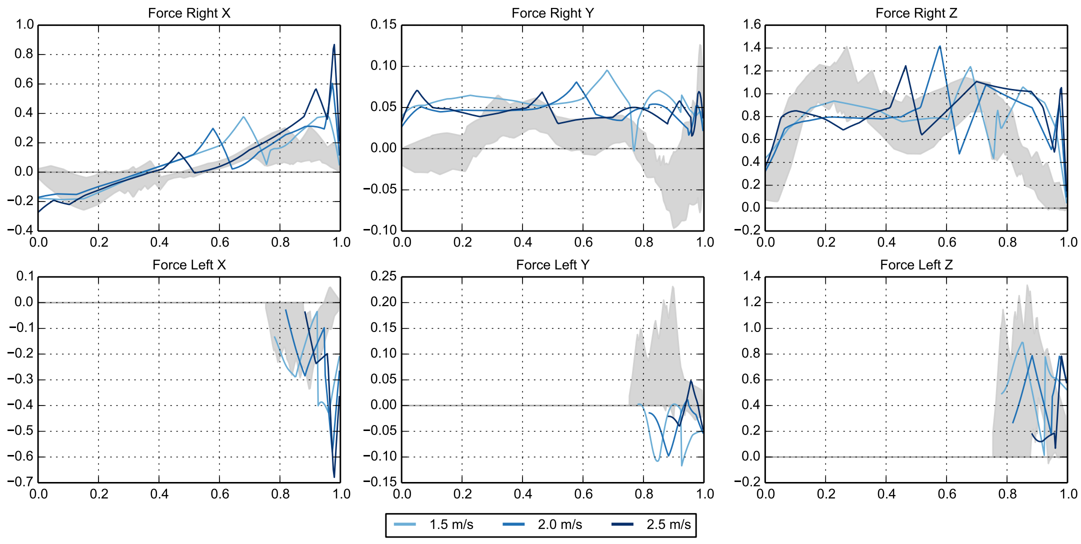


(b) Elevation angles of the synthesized gaits for different walking velocities.

Figure 6.13: Elevation angles of the different gait synthesis formulations.



(a) Weight normalized ground reaction forces of the synthesized gaits for the different objective functions.



(b) Elevation angles of the synthesized gaits for different walking velocities.

Figure 6.14: Weight normalized ground reaction forces of the different gait synthesis formulations. The ground reaction forces of the reconstructed gaits are shown as grey areas in the background.

Basic Objective Function Contributions of the Synthesized Gaits

To assess the influence of the basic objective functions to each other we cross-evaluated the objective functions. In Figure 6.15 we compare the evaluations of the five basic objective functions $\Phi_{L_1}, \Phi_{L_2}, \Phi_{L_3}, \Phi_{M_4}$ using the optimal solutions of the synthesized gaits that minimized w.r.t. $\Phi_1, \ldots, \Phi_4$. For each of the objective functions we used the obtained optimal solutions $\boldsymbol{x}(t)^*, \boldsymbol{u}(t)^*, \boldsymbol{p}^*$ to cross-evaluate the basic objective functions. Numerical values of these evaluations are presented in Table 6.6.

As expected for each of the four objective functions always the optimal solution of the corresponding criteria yields the smallest value, i.e. the solution using *Minimize Actuation over Step Length* has the smallest objective value for the basic objective function Φ_{L_1}, using *Minimize Angular Momentum* has the smallest objective value for the basic objective functions Φ_{L_2}, etc.

The objective functions Φ_{L_1} (minimize active actuation over step length) and Φ_{L_2} (minimize angular momentum) appear to be independent of each other as the plots of their evaluations show a strong reduction of the objective value only when minimizing those criteria specifically. Using *Minimize Angular Momentum* results in similar values for the Φ_{L_1} criterion as *Minimize Head Angular Velocity* and *Minimize Step Duration*. Similarly when optimizing for *Minimize Actuation over Step Length* one obtains similar values for the Φ_{L_2} criterion as *Minimize Head Angular Velocity* and *Minimize Step Duration*.

One can notice that minimizing the head angular velocity also results in a rather short step duration. The reason for this is that minimizing the head angular velocity reaches a very small value already as the model has enough degrees of freedom to to keep it near zero. The only way to further reduce the objective function is then to shorten the time interval over which the Lagrange term of Φ_3 is evaluated.

		Evaluated Objective Function			
		Φ_{L_1}	Φ_{L_2}	Φ_{L_3}	Φ_{M_4}
Optimal Solution using	Φ_1	8.19×10^{-2}	$5.62 \times 10^{+0}$	1.54×10^{-1}	4.84×10^{-1}
	Φ_2	8.45×10^{-1}	8.28×10^{-4}	3.57×10^{-1}	3.89×10^{-1}
	Φ_3	8.96×10^{-1}	$5.03 \times 10^{+0}$	6.39×10^{-5}	2.65×10^{-1}
	Φ_4	$1.07 \times 10^{+0}$	$4.22 \times 10^{+0}$	1.28×10^{-1}	2.00×10^{-1}
	Φ_5	1.86×10^{-1}	1.16×10^{-2}	3.34×10^{-4}	5.04×10^{-1}

Table 6.6: Numerical values of the objective function cross evaluations.

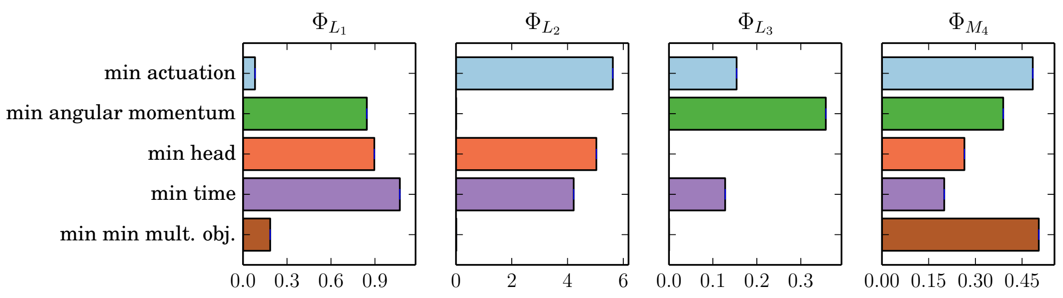


Figure 6.15: Cross-evaluation of the objective functions for the obtained optimal solutions. The title of each plot shows the name of the evaluated objective function and each bar is the optimal solution we used to evaluate the objective function.

Basic Objective Function Contributions of the Reconstructed Gaits

We also evaluated the objective functions for the reconstructed gait dynamics. The results are shown in Figure 6.16. As the model did not use any spring-dampers we used 0 for all spring-damper parameters when evaluating the objective functions.

The plots are shown in the same manner as for the cross evaluation of the objective functions in that the title of a plot is the evaluated objective function and the different bars represent the reconstructed gaits that were used as inputs for the evaluation. As we have three trials per emotional expression we show the average value as the bar together with indications for the standard deviation as blue error bars.

The objective function Φ_{L_1} (minimize active actuation over step length) can be seen as a measure of effort used to produce a step and an interesting observation can be made by comparing the its evaluation with the average walk velocities shown in Figure 6.2: the expression of *JoyExpressive* has a similar level of effort to perform the step to the expression of *AngerSupressive*, however the average walk velocity of *JoyExpressive* are higher than those of *AngerSupressive*.

The evaluations of Φ_{L_2} (minimize centroidal angular momentum) show higher values for the expressions of angry movements and the lowest value is obtained for *JoyExpressive*. However *JoySelfsufficient* has a higher average value than the expression *Neutral*. Overall the values are of similar magnitude as the synthesized motions where the angular momentum was not minimized. This could indicate that the angular momentum in human is kept at moderately low values due to the intrinsic motion itself.

The third plot from the left shows the evaluations of the basic objective function Φ_{L_3} (minimize head angular velocity). The magnitudes obtained here are relatively small. The expression of *Neutral* shows the smallest values and the average value for *AngerExpressive* is smaller than that of *AngerSupressive*. Similarly the head angular velocity of *JoyExpressive* is smaller than that of *JoySupressive*.

The evaluations of *Minimize Step Duration* simply restate the values shown in Figure 6.16.

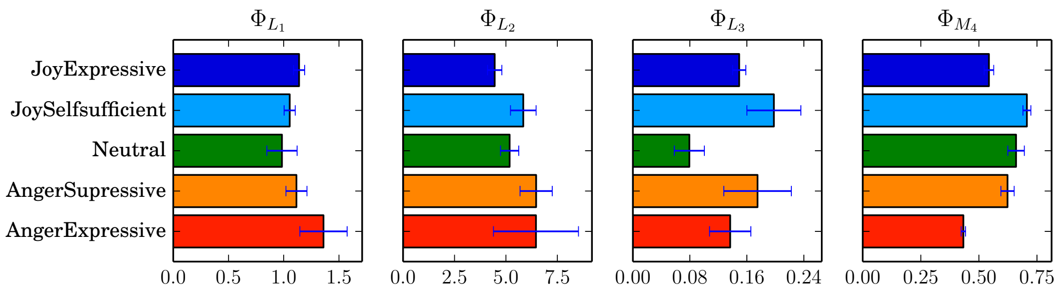


Figure 6.16: Comparison of objective function values for the reconstructed gaits.

6.5 Summary

In this chapter we have presented the numerical results of this thesis. We presented the subject-specific models and how they allow us to reproduce the motion capture marker data using proper rigid multibody motions. We listed all parameters that we use as input parameters for our HeiMan model to create the subject-specific dynamic models.

We analyzed the recorded emotional gaits on a kinematic level using standard gait parameters and also by looking at the joint kinematics. Especially joints of shoulder and elbows showed features that we assume to be emotion specific.

We presented the results of our dynamic gait reconstruction and by analyzing the fit error we can conclude that we can closely reproduce the recorded kinematic motions even with a dynamic model that is subject to additional kinematic contact constraints. We analyzed the reconstructed emotional motions using the elevation angles, joint forces, and ground reaction forces. We further compared the reconstructed joint forces with the recorded EMG signals and found emotion-specific features that are present in both joint forces and EMG signals.

We described the synthesized gaits that we generated by using different objective functions or prescribing different walking velocities. We presented a similarity measure and used it to compare the synthesized motions with the reconstructed motions. We studied the obtained optimal synthesized gaits using elevation angles and ground reaction forces. Further we analyzed the evaluations of the basic objective functions on the synthesized walking motions and the reconstructed emotional gaits.

Our results show that a wide range of motions with different characteristics can be created by adjusting weightings in the objective or changing gait parameters. Further our results show that similarity to recorded motions can be improved when using different combinations of the basic objective functions.

Chapter 7

Summary and Outlook

Summary

In this thesis we have approached the study of emotional body language from a completely new direction. Central components of our approach are the use of dynamics of rigid multibody modeling and numerical optimal control methods, which we combine to create a new mathematical model for 3-D full-body human walking motions as an optimally controlled process. We successfully applied our formulation to the analysis of emotional walking data and are able to create open-loop human-like walking motions without the use of any reference data, solely based on mathematical principles. To achieve our goals we have made contributions in multiple areas:

A New Method to Transfer Motion Capture Data onto Rigid Multibody Models

The motion capture data consists of marker data that are attached to specific parts of the subject and are subject to non-rigid movements such as skin and muscle movement. The markers are further redundant, which makes assessment of the underlying skeletal movement difficult. To address this we created HeiMan, a detailed 3-D rigid multibody model to describe the movement of the human body. It is highly parameterized to allow adjustments to closely represent a recorded subject. We employ a robust inverse kinematics method based on non-linear programming to transfer the motion capture data onto these subject specific models. In doing so we are able to reproduce recorded motion capture marker movement using proper rigid multibody models with high precision and a set of independent coordinates that describe the movement of the model. We implemented all methods in the motion capture mapping tool Puppeteer.

A New High Performance Rigid Body Dynamics Implementation

We created a highly efficient rigid multibody dynamics implementation named RBDL that is specifically tailored for use in the context of numerical optimal control. It uses recursive algorithms and is based on Featherstone's 6-D spatial algebra to formulate rigid body movement. High performance is achieved by exploiting algebraic structures and sparsities in both spatial algebra and quantities in the equation of motion for rigid multibody system with algebraic contact constraints expressed in reduced coordinates. It allows to selectively reuse already computed values to surpass performance that is otherwise only available using symbolically generated code, while being much more flexible to use. We created more than 180 tests to ensure correctness of individual parts of the implementation. Its performance and the model-based programming interface allows reusing of all implemented algorithms also in other areas such as robotic control or virtual characters.

Modeling of Human Walking as a Nonlinear Multi-Phase Optimal Control Problem

We created a new model for the analysis of human walking by formulating it as a non-linear multi-phase optimal control problem. It uses the full-body subject specific models to describe the dynamics of the human model. Each phase of the optimal control problem describes the dynamics of the model under different nonlinear contact constraints. We created two formulations: a dynamic reconstruction formulation and a gait synthesis formulation.

The dynamic reconstruction uses a least-squares type objective function to approximate a given motion sequence to reconstruct joint torques and ground reaction forces from purely kinematic reference motions. The gait synthesis formulation on the other hand allows generation of full-body open-loop 3-D walking movements without the use of any reference data. It enables a new way to study fundamental principles of 3-D full-body human walking by formulating them as optimality criteria, constraints, or model parameters.

Reconstruction of Emotional Aspects in the Dynamics of Human Walking

We have applied our dynamic reconstruction formulation on 15 motion capture sequences from our experiments. The reconstructed joint torques show characteristics that are also found in the recorded EMG data such as the increased efforts for the expressions of *AngerExpressive* compared to the other emotions. But also similarities for individual muscles and joint actuations could be found. Our reconstruction is performed on a 3-D full-body model and therefore allows the study of effects of upper body movements that add to quantities of standard gait analysis procedures.

Our approach allows the study of expressions and motor control of human movement from a new perspective, without having to integrate additional measurements such as force plates into the experiment. Especially for studying emotional expressions this is a great benefit as subjects are given greater freedom in their movement without having to place their feet on specific areas on the ground.

To our knowledge we are the first to investigate emotional expressions using a dynamic 3-D full-body model. Our results suggest that actuations and muscular activity in the upper body play an important role in the expressions of emotions and that also movement in the frontal plane, especially abduction of the shoulder, should not be neglected.

A New Database of Verified Emotional Motion Data

We have conducted a motion capture experiment with five subjects to obtain data that contains both the kinematic motion description and muscle activity of both upper and lower body in form of electromyographic data. We have created an online survey with more than one hundred participants to evaluate the expressiveness of the recorded motions. The results show large differences in how well the emotions were recognized. Nevertheless we were able to identify recordings for which emotions were very well recognized and that we analyzed with the methods developed for this thesis.

Outlook

The results of this thesis strongly support the use of full-body rigid multibody models in the study of emotional expressions and gait analysis in general. The created models and methods can be used in a wide range of new research efforts, such as:

- **Investigations of Synergies in Joint Torques**
  Research in gait analysis has found synergies of the elevation angles in foot, shank, and ankle, but also in EMG signals during walking. Investigation of synergies in the joint torques and how they are affected by different emotions could give new insights in the motor control of emotional expressions. A further application of these synergies could be used to formulate controllers that only require few input signals but are capable of driving actuated and emotionally expressive full-body virtual characters.

- **Investigation of Principles of Human Walking**
  Our synthesis model allows the investigation of human walking in a new way: formulating hypothesises of basic principles as optimality conditions, constraints, or model parameter can be easily tested using our full-body 3-D gait model. Examples would be the investigation

of the role of arms by computing gaits with immobilized arms or replacement of limbs by completely passive elements to model prostheses. Also different principles of stability could be formulated and tested with our model.

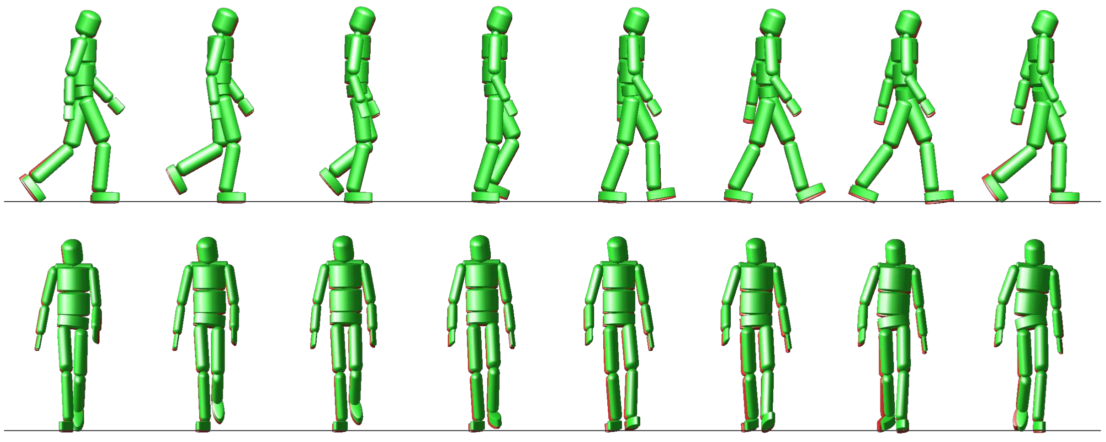
- **Integration of Muscle Models**
  The model we developed in this thesis describes all actuations directly at the joints. These joint torques can be used to estimate muscular activity. This would help to further validate our model and further open up new possibilities for analyzing human movements on a muscular level that however do not need force plates or EMG recordings but compute these informations solely from kinematic recordings.

- **Inverse Optimal Control of Human Walking**
  Our gait synthesis formulation is ideally suited to perform inverse optimal control to identify optimality criteria that govern human walking. Our gait synthesis model will play a crucial component in such a computation. Furthermore the motion capture data from our experiments can be used to identify criteria of emotional expressions.

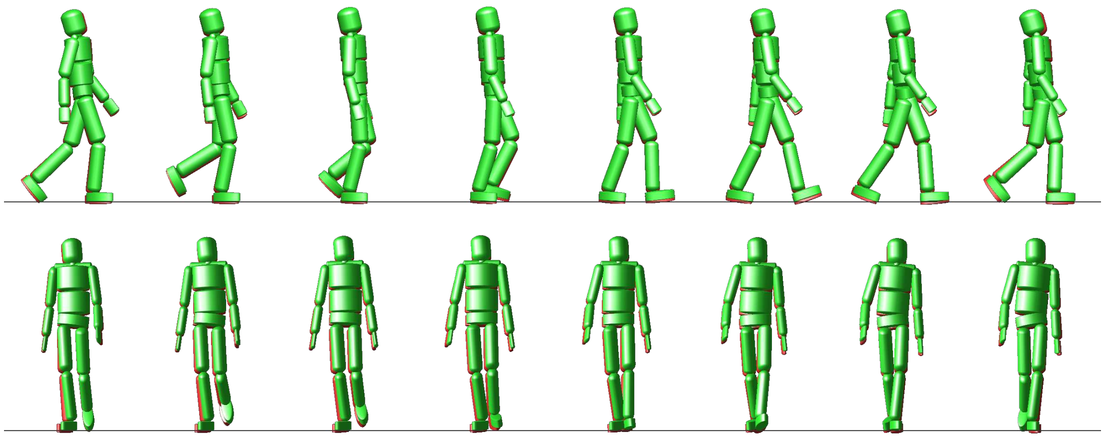
Appendix A

Visualizations and Trajectories

A.1 Visualization of the Dynamic Fit Error



(a) *AngerSupressive*



(b) *JoySelfsufficient*

Figure A.1: Visualization of the movements and error between the movement after the dynamic LSQ fit (shown in green) and reference movement from motion capture (shown in red) for *AngerSupressive* and *JoySelfsufficient*. Other expressions were shown in Figure 6.5.

A.2 Reconstructed Trajectories

AngerExpressive: Generalized Positions

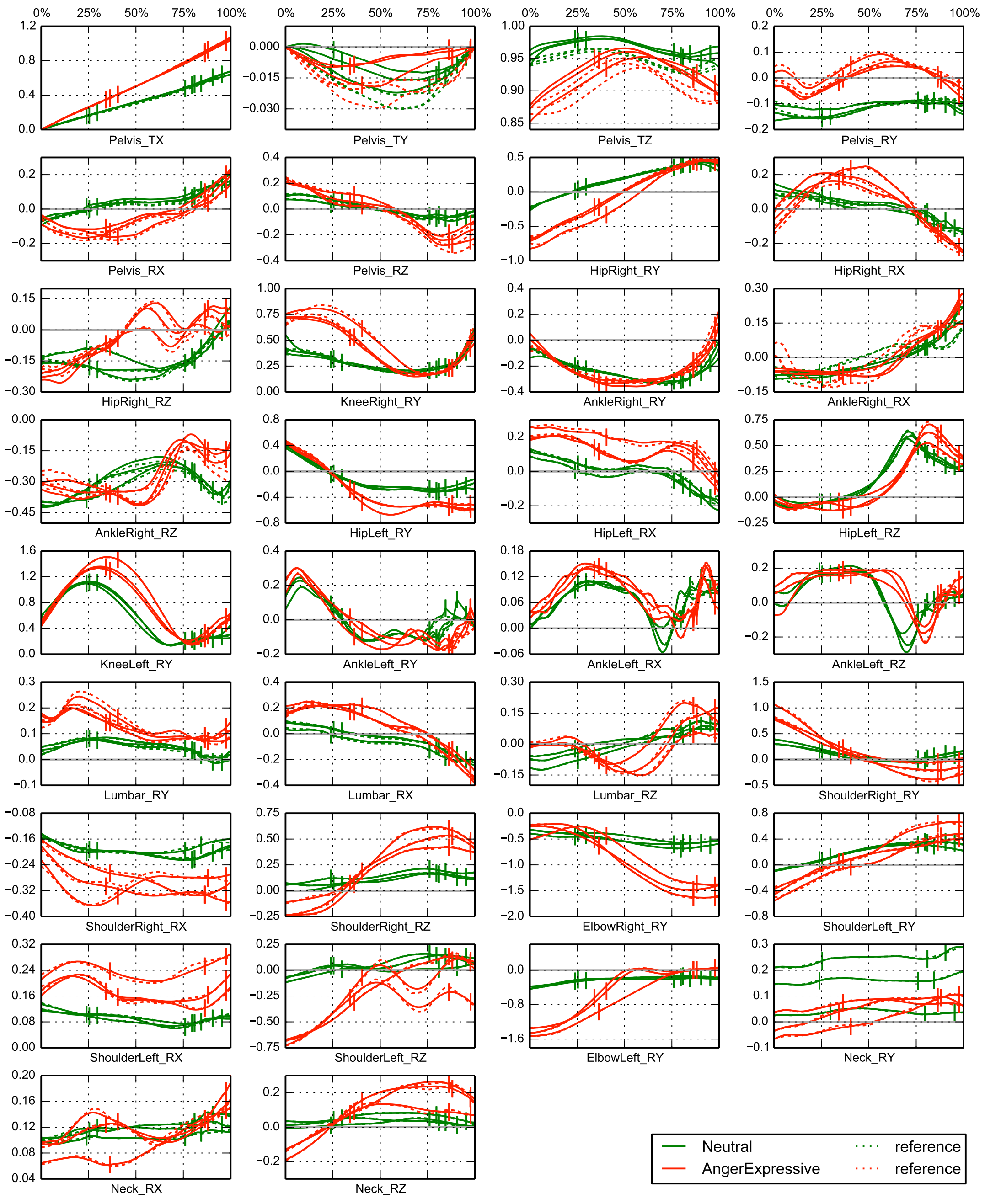


Figure A.2: Model generalized joint trajectories and reference data for expressions of *Neutral* and *AngerExpressive*. Vertical markers indicate the phase transitions of the gaits.

***AngerExpressive*: Joint Torques**

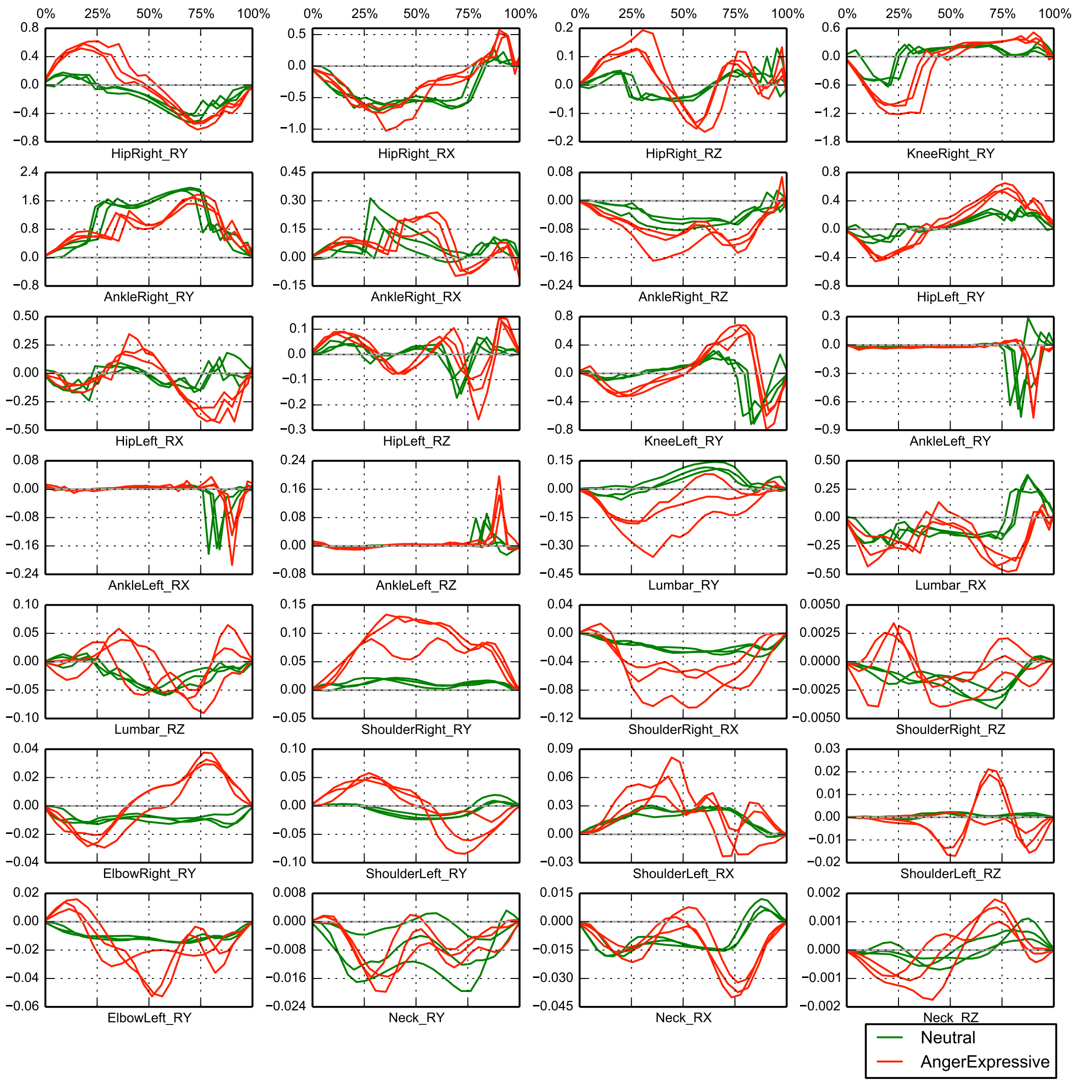


Figure A.3: Weight normalized joint torques for the reconstructed expressions of *Neutral* and *AngerExpressive*.

***AngerSupressive*: Generalized Positions**

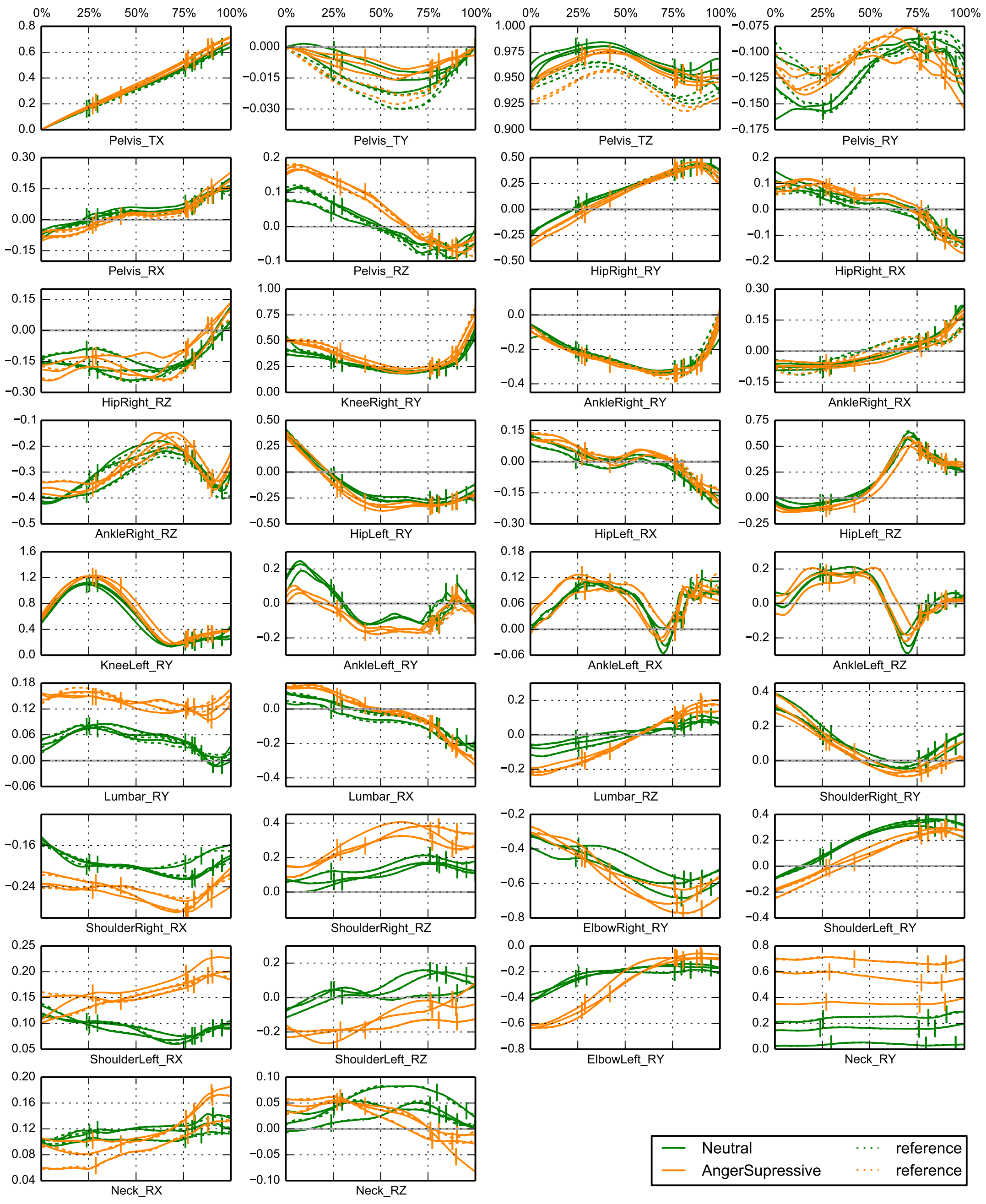


Figure A.4: Model joint trajectories and reference data for expressions of *Neutral* and *AngerSupressive*. Vertical markers indicate the phase transitions of the gaits.

***AngerSupressive*: Joint Torques**

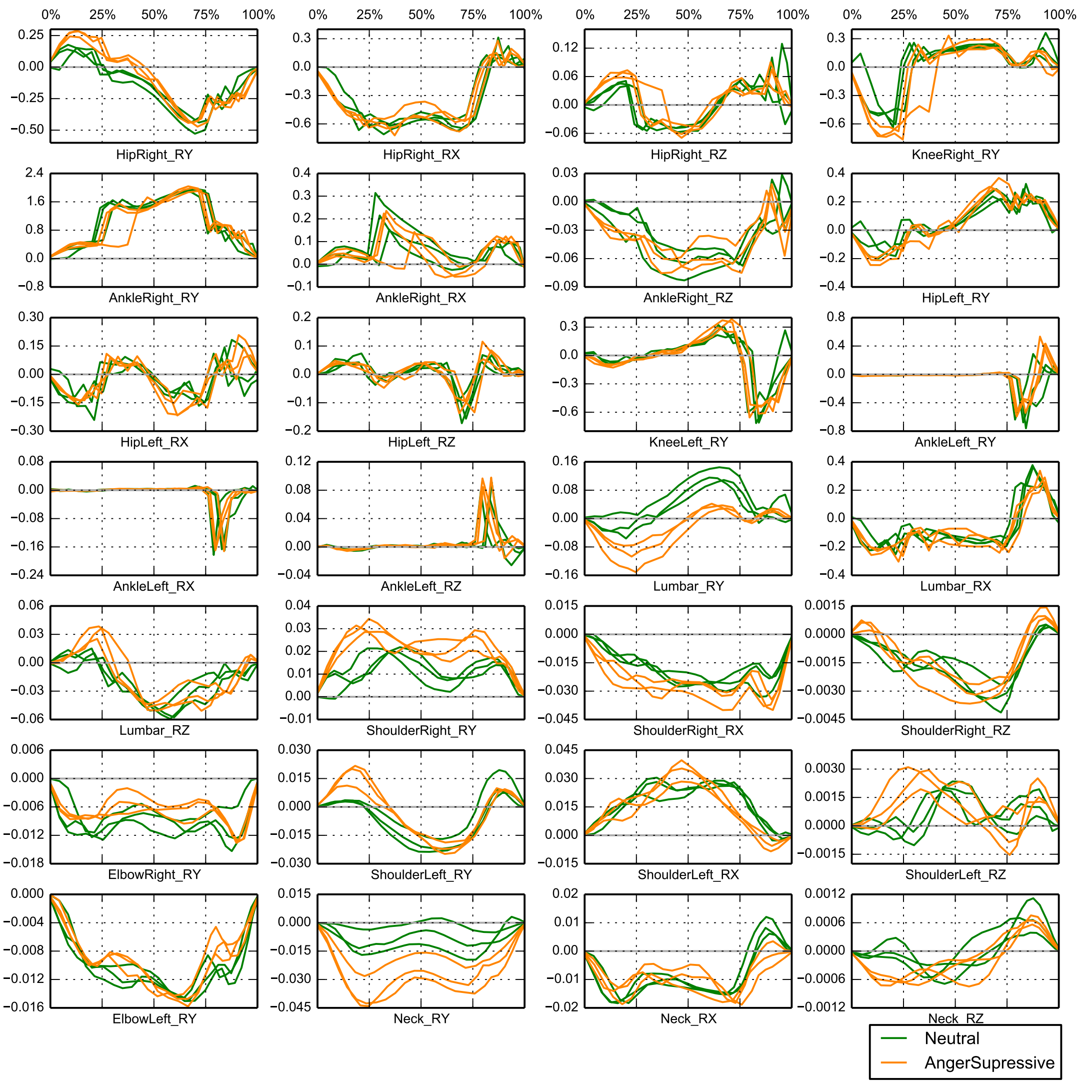


Figure A.5: Weight normalized joint torques for reconstructed expressions of *Neutral* and *AngerSupressive*.

JoySelfsufficient: Generalized Positions

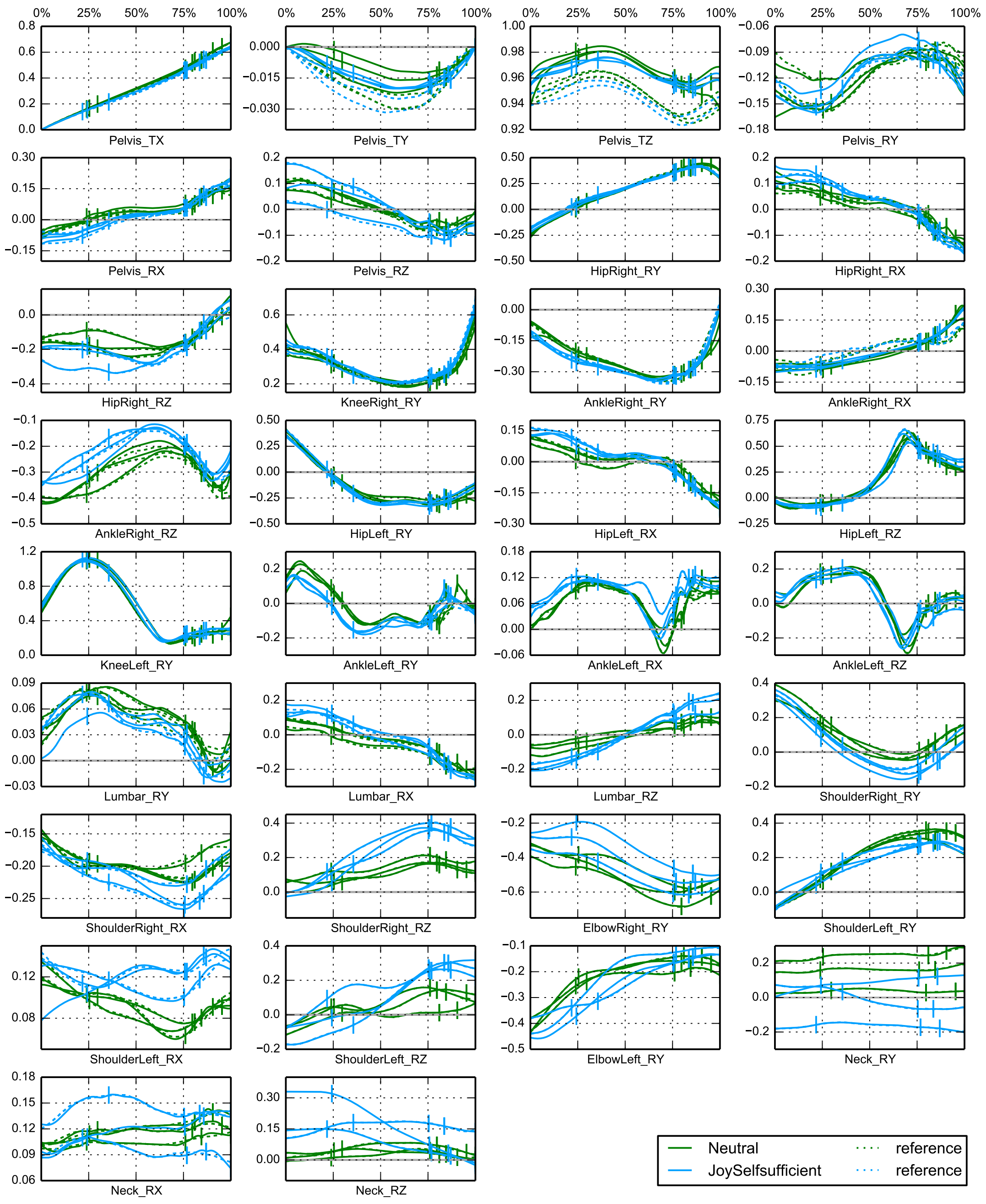


Figure A.6: Model joint trajectories and reference data for expressions of *Neutral* and *JoySelfsufficient*. Vertical markers indicate the phase transitions of the gaits.

***JoySelfsufficient*: Joint Torques**

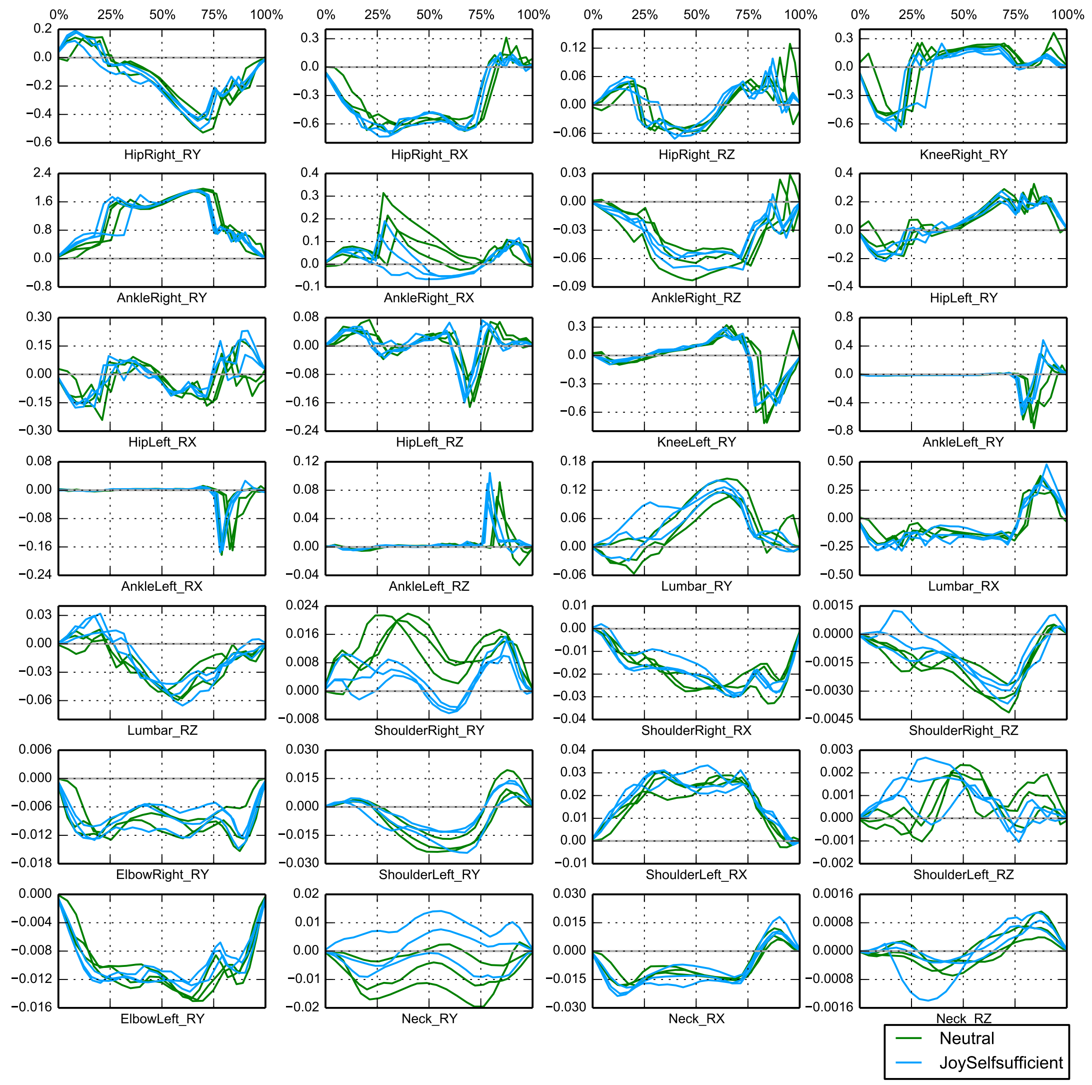


Figure A.7: Weight normalized joint torques for the reconstructed expressions of *Neutral* and *JoySelfsufficient*.

JoyExpressive: Generalized Positions

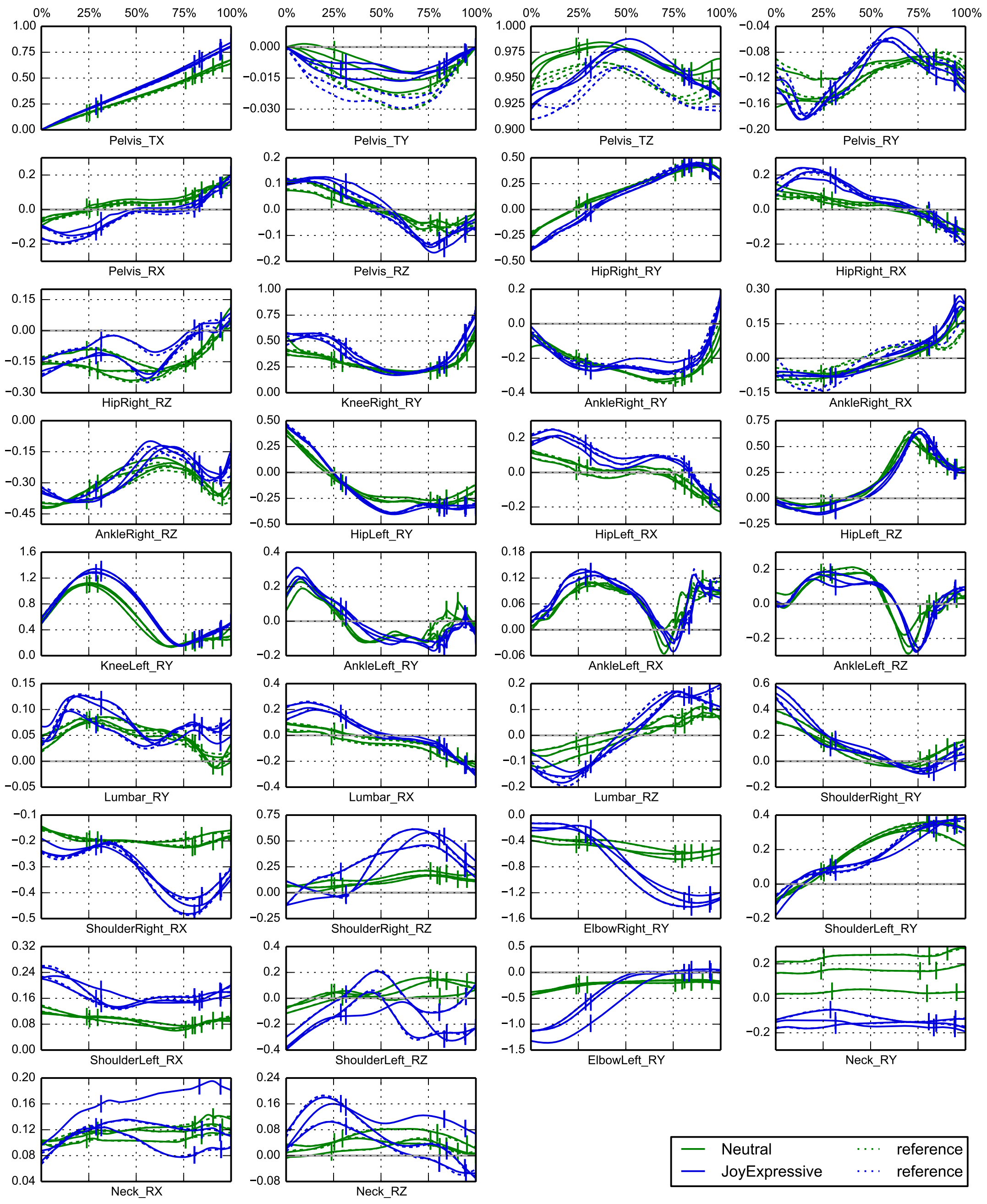


Figure A.8: Model joint trajectories and reference data for expressions of *Neutral* and *JoyExpressive*. Vertical markers indicate the phase transitions of the gaits.

***JoyExpressive*: Joint Torques**

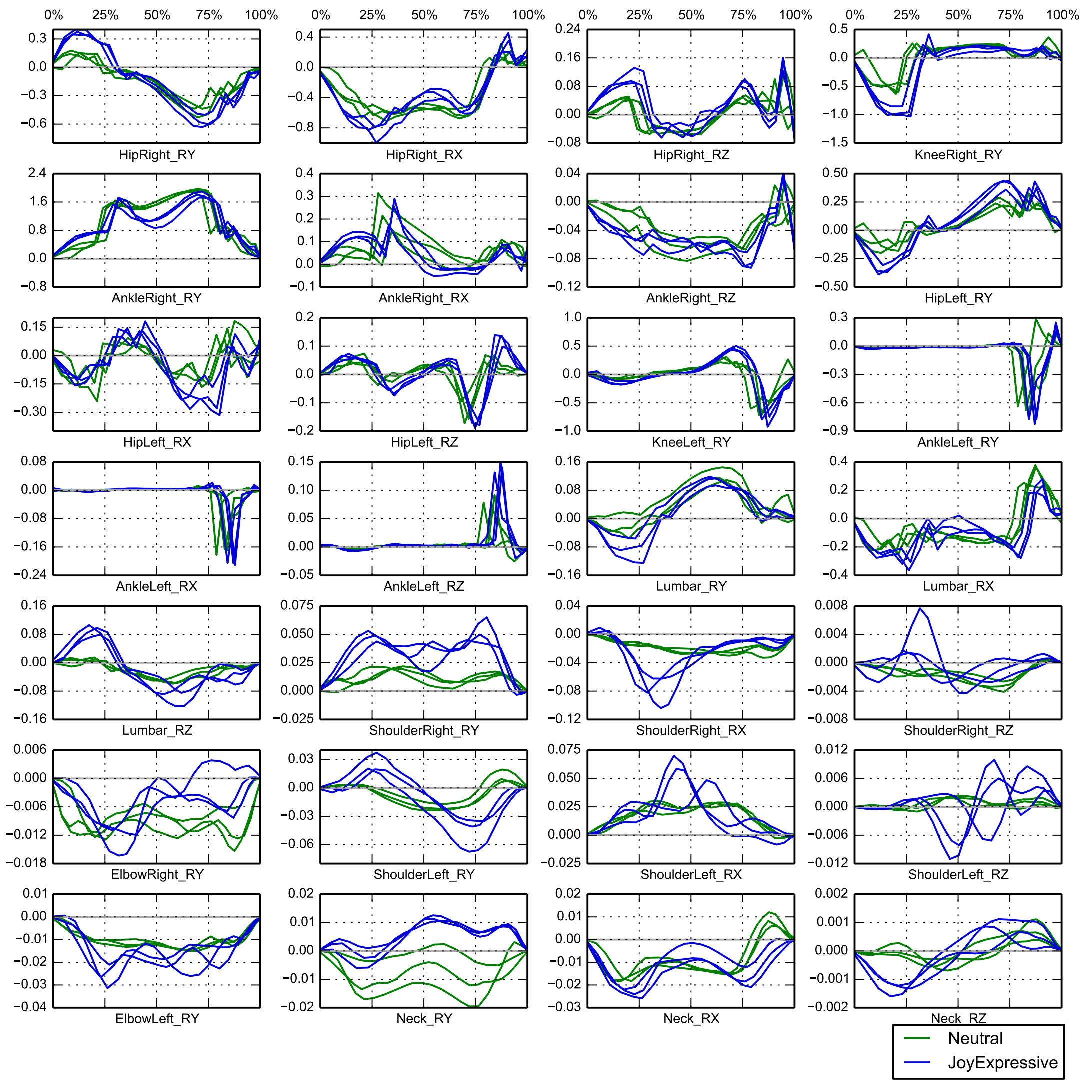


Figure A.9: Weight normalized joint torques for the reconstructed expressions of *Neutral* and *JoyExpressive*.

A.3 Trajectories of the Synthesized Gaits: Objective Function Variations

Generalized Positions

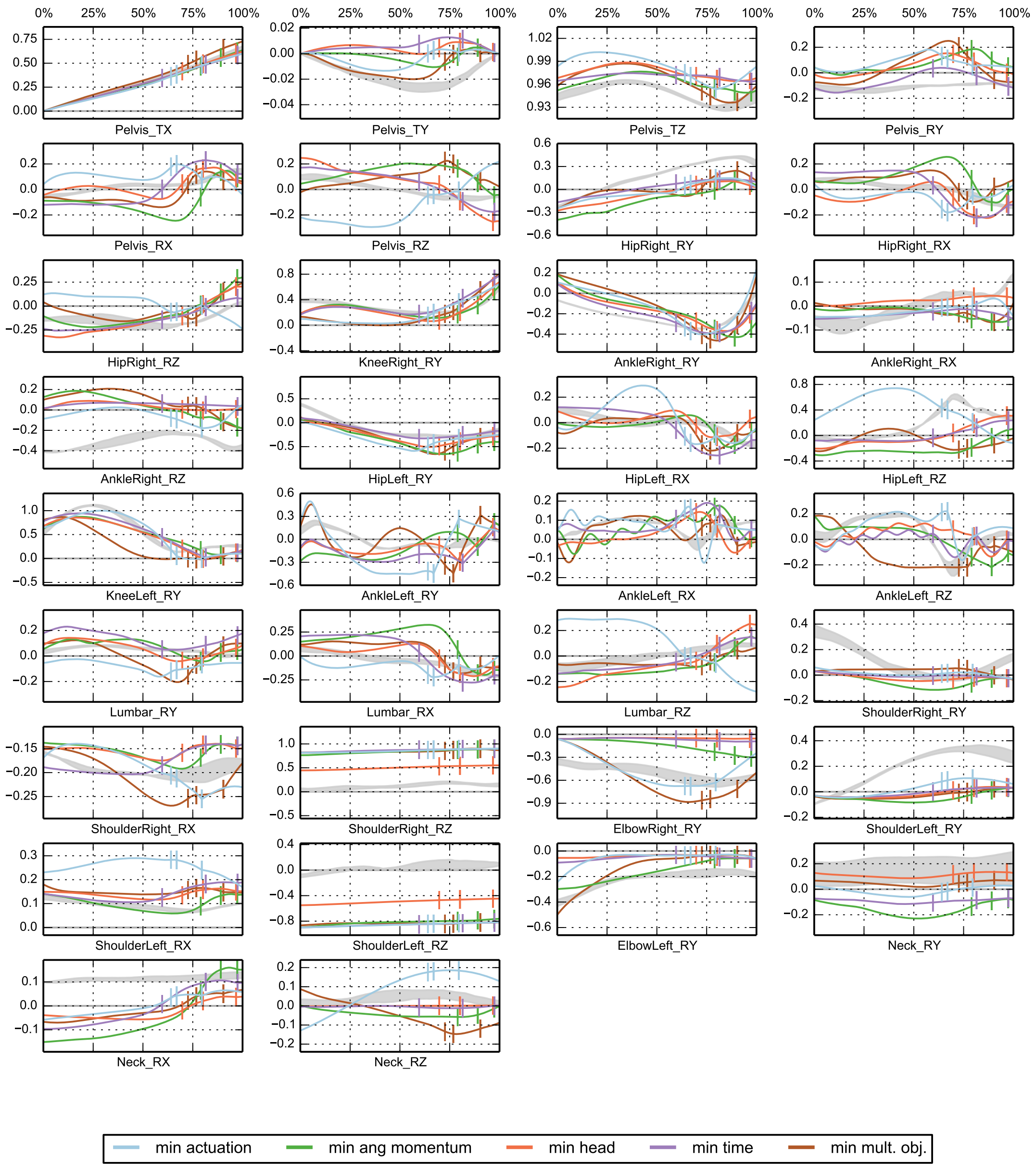


Figure A.10: Joint trajectories for gaits that we synthesized using different objective criteria. Vertical markers indicate the phase transitions of the gaits. Areas shaded in grey show the generalized joint positions of the *Neutral* expressions as reference.

Generalized Joint Torques

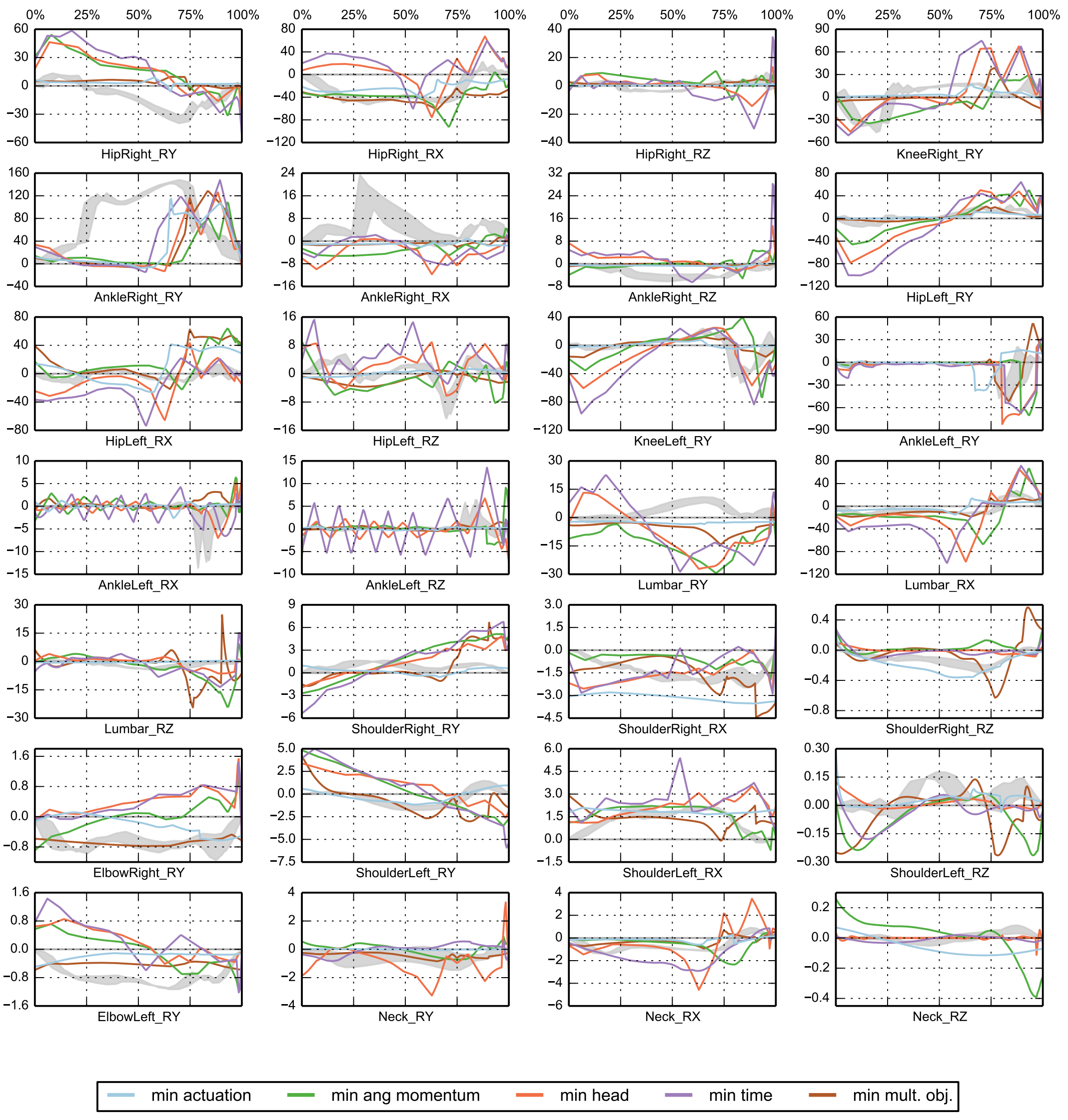


Figure A.11: Weight normalized joint torques for gaits that we synthesized using different objective criteria. Areas shaded in grey show the generalized joint torques of the reconstructed *Neutral* expressions as reference. The joint torques shown here include both the forces of the active actuation $\boldsymbol{u}(t)$ and the passive actuations of the spring-dampers. Contributions of the latter are shown in Figure A.12.

Spring-Damper Contributions

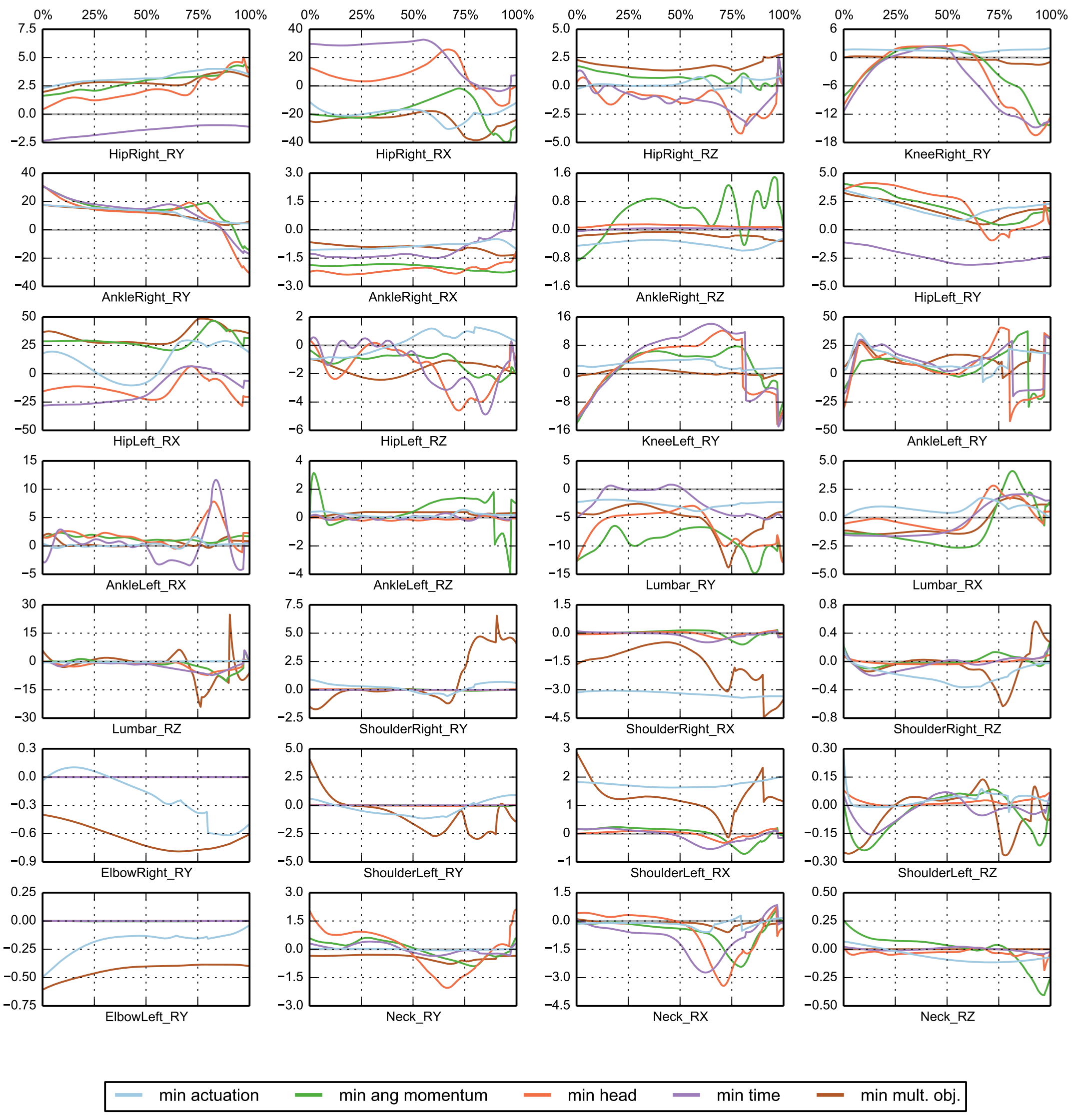


Figure A.12: Weight normalized joint torque contributions of the spring-damper forces for the synthesized gaits using different objective criteria. Here we only show the passive forces resulting from the spring-damper elements at the joints. Plots of the full joint forces that include both active and passive contributions are shown in Figure A.11.

Spring-Damper Parameters

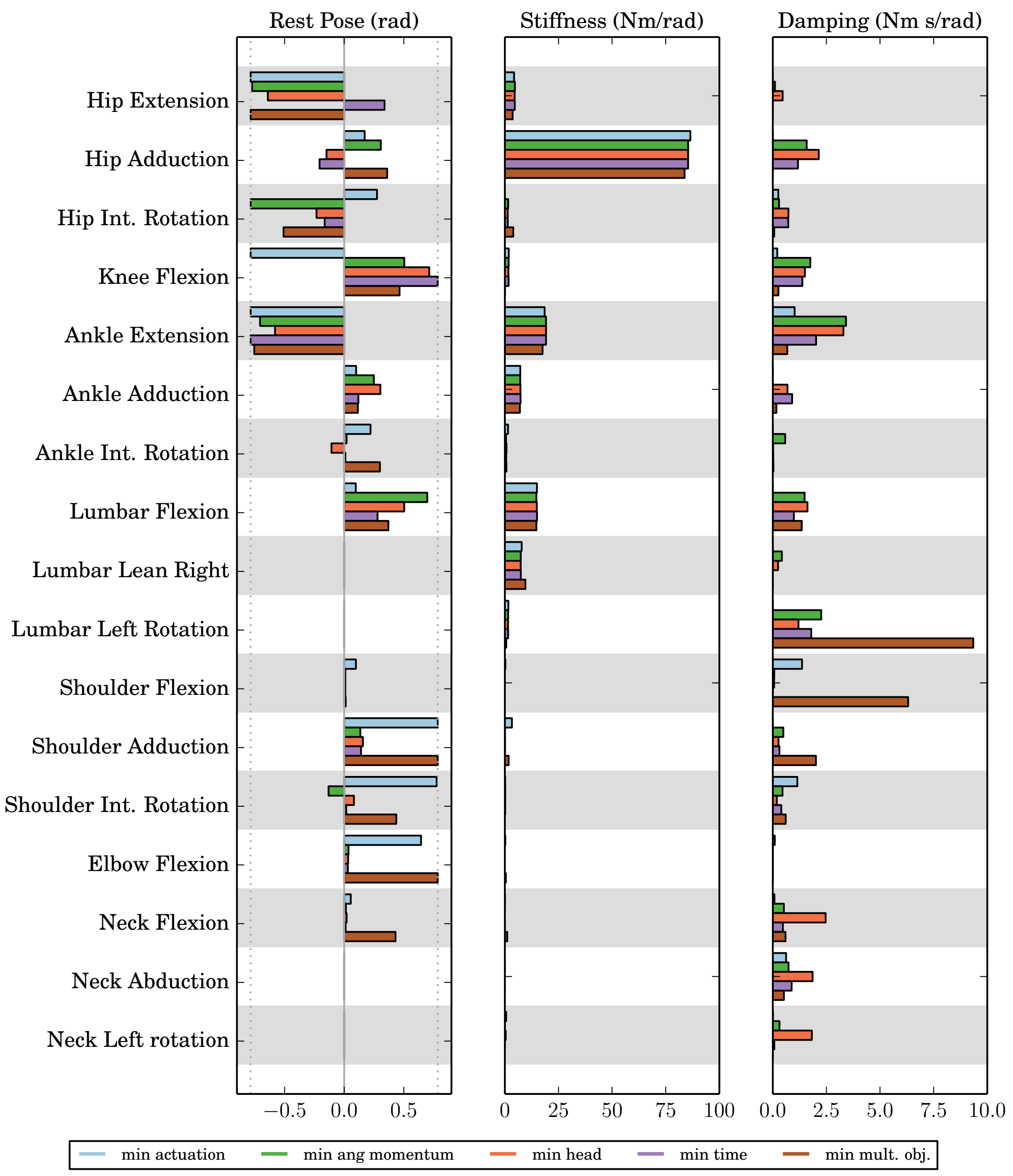


Figure A.13: Spring-damper parameters that we obtained by solving the gait synthesis formulation using the different objective functions. The dashed vertical lines for in the rest pose plots show upper and lower bounds for the parameters.

A.4 Trajectories of the Synthesized Gaits: Walking Velocity Variations

Generalized Positions

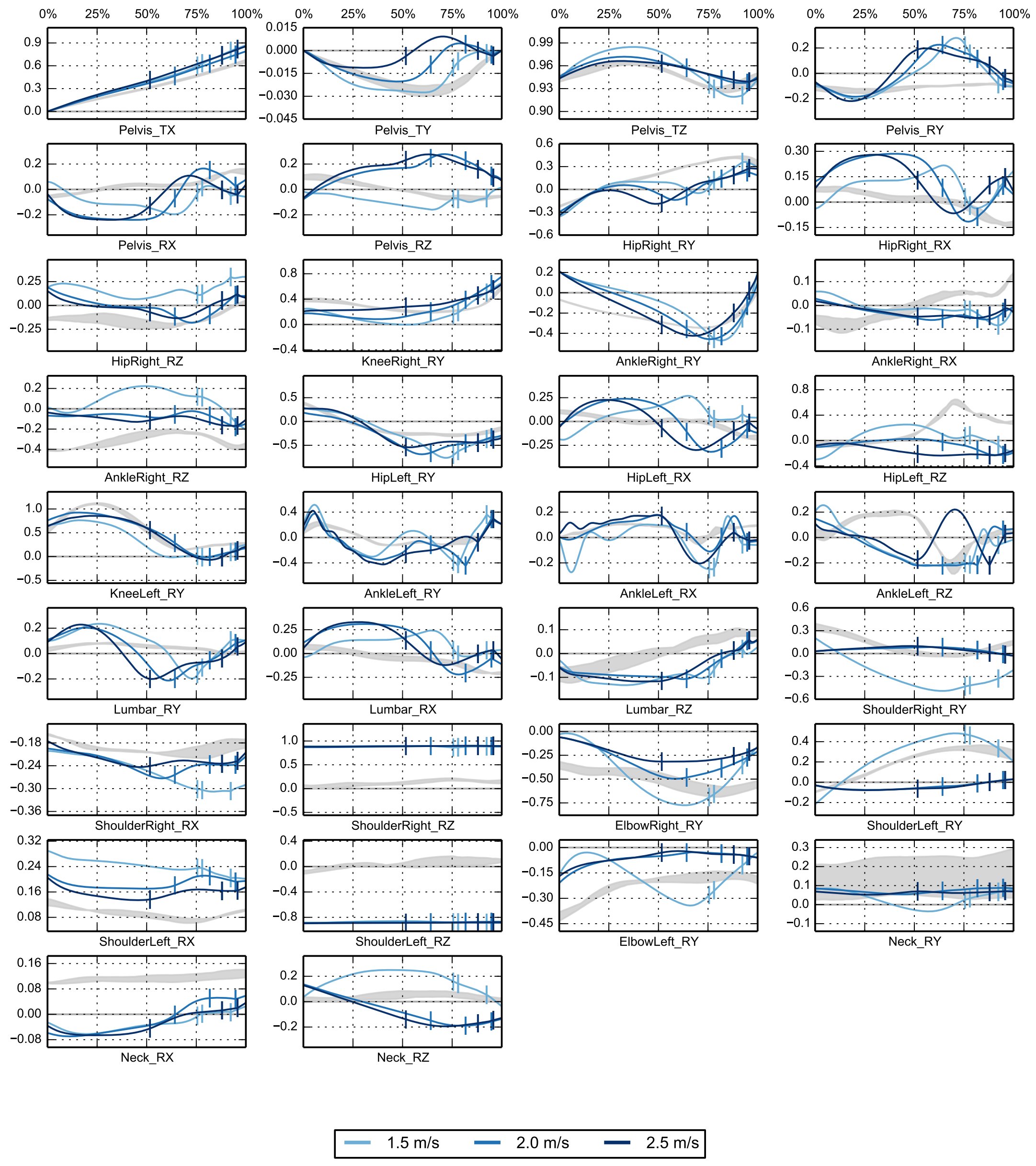


Figure A.14: Joint trajectories for gaits that we synthesized using the *Multi Objective* criterion and different prescribed average walking velocities. Vertical markers indicate the phase transitions of the gaits. Areas shaded in grey show the generalized joint positions of the *Neutral* expressions as reference.

Generalized Joint Torques

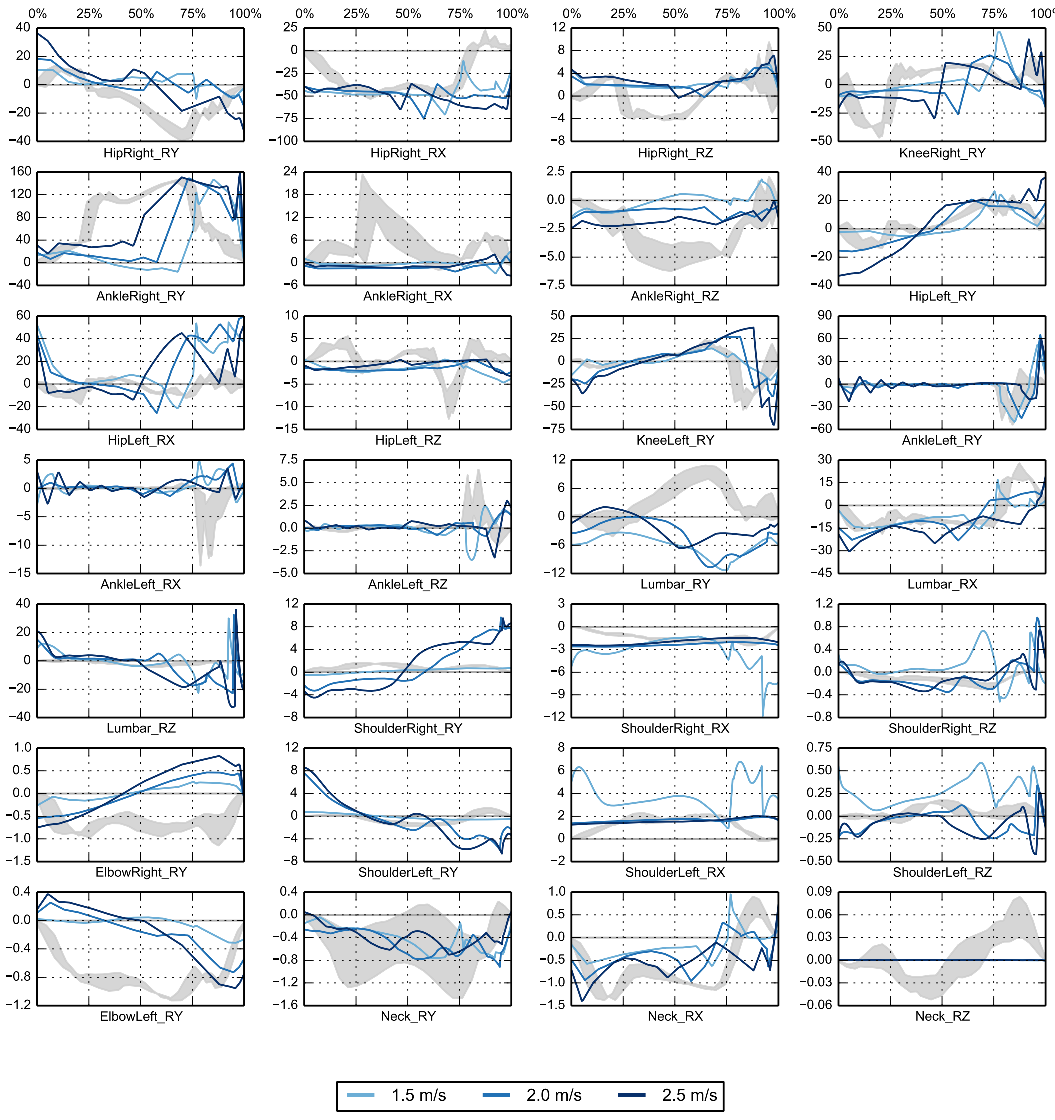


Figure A.15: Weight normalized joint torques for gaits that we synthesized using the *Multi Objective* criterion and different prescribed average walking velocities. Areas shaded in grey show the torques of the reconstructed *Neutral* expressions as reference. The joint torques shown here include both the forces of the active actuation $\boldsymbol{u}(t)$ and the passive actuations of the spring-dampers. Contributions of the latter are shown in Figure A.16.

Spring-Damper Contributions

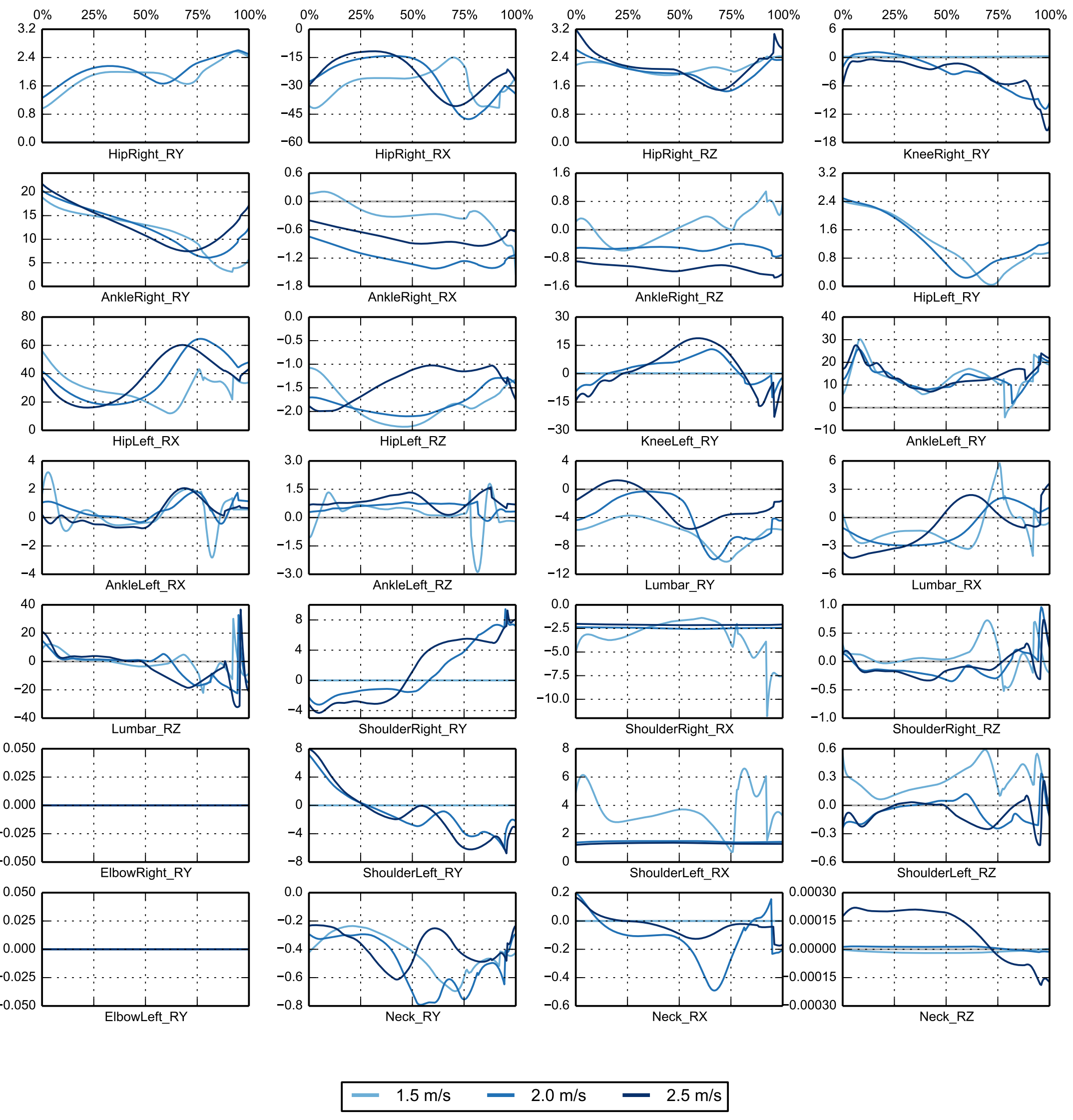


Figure A.16: Weight normalized joint torque contributions of the spring-damper forces for the synthesized gaits using the *Multi Objective* criterion and different prescribed average walking velocities. Here we only show the passive forces resulting from the spring-damper elements at the joints. Plots of the full joint forces that include both active and passive contributions are shown in Figure A.15.

Spring-Damper Parameters

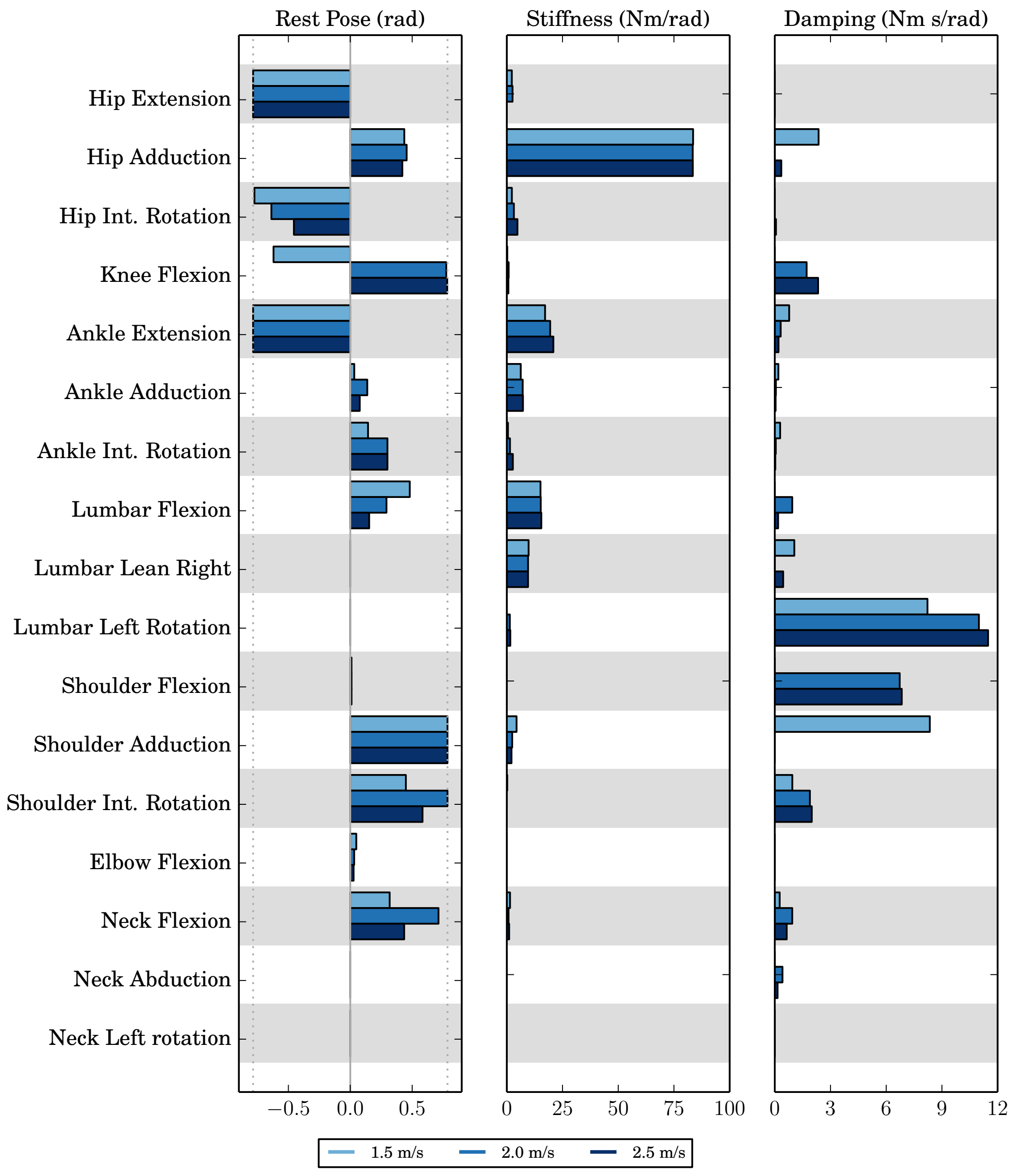


Figure A.17: Spring-damper parameters that we obtained by solving the gait synthesis formulation using prescribed average walking velocities. The dashed vertical lines for in the rest pose plots show upper and lower bounds for the parameters.

Bibliography

Alexander, R. M. (1984). The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research*, 3(2):49–59. (p. 2.)

Alexander, R. M. (1997). A minimum energy cost hypothesis for human arm trajectories. *Biological cybernetics*, 76(2):97–105. (p. 2.)

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., and Sorensen, D. (1999). *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition. (p. 33.)

Ascher, U. M., Chin, H., Petzold, L. R., and Reich, S. (1995). Stabilization of constrained mechanical systems with daes and invariant manifolds. *Journal of Structural Mechanics*, 23(2):135–157. (p. 32.)

Atkinson, A. P., Dittrich, W. H., Gemmell, A. J., Young, A. W., et al. (2004). Emotion perception from dynamic and static body expressions in point-light and full-light displays. *Perception*, 33(6):717–746. (p. 1 and 14.)

Aviezer, H., Trope, Y., and Todorov, A. (2012). Body cues, not facial expressions, discriminate between intense positive and negative emotions. *Science*, 338(6111):1225–1229. (p. 1 and 14.)

Barliya, A., Omlor, L., Giese, M., Berthoz, A., and Flash, T. (2013). Expression of emotion in the kinematics of locomotion. *Experimental Brain Research*, 225(2):159–176. (p. 15.)

Barry, P. J. and Goldman, R. N. (1988). A recursive evaluation algorithm for a class of catmull-rom splines. *SIGGRAPH Comput. Graph.*, 22(4):199–204. (p. 86.)

Berthoz, A. (2003). *Emotion and Reason*. Oxford University Press. (p. 1.)

Blickhan, R. (1989). The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11–12):1217 – 1227. (p. 1.)

Bock, H. (1981a). Numerical treatment of inverse problems in chemical reaction kinetics. In Ebert, K., Deuflhard, P., and Jäger, W., editors, *Modelling of Chemical Reaction Systems*, volume 18 of *Springer Series in Chemical Physics*, pages 102–125. Springer, Heidelberg. (p. 84.)

Bock, H. (1981b). Numerische Behandlung von zustandsbeschränkten und Chebyshev-Steuerungsproblemen. Technical Report R106/81/11, Heidelberg. (p. 79.)

Bock, H. and Plitt, K. (1984). A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, pages 243–247. Pergamon Press. (p. 2, 79, and 83.)

Borghese, N. A., Bianchi, L., and Lacquaniti, F. (1996). Kinematic determinants of human locomotion. *The Journal of Physiology*, 494(Pt 3):863–879. (p. 13 and 108.)

Brownlow, S., Dixon, A. R., Egbert, C. A., and Radcliffe, R. D. (1997). Perception of movement and dancer characteristics from point-light displays of dance. *The Psychological Record*, 47(3):411–421. (p. 14.)

Chan, S. and Lawrence, P. (1988). General inverse kinematics with the error damped pseudoinverse. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 834–839 vol.2. (p. 71.)

Collins, S. H., Adamczyk, P. G., and Kuo, A. D. (2009). Dynamic arm swinging in human walking. *Proceedings of the Royal Society B: Biological Sciences*, pages –. (p. 13.)

Craig, J. (2005). *Introduction to Robotics: Mechanics and Control*. Addison-Wesley series in electrical and computer engineering: control engineering. Pearson Education, Incorporated. (p. 29.)

Cutting, J. and Kozlowski, L. (1977). Recognizing friends by their walk: Gait perception without familiarity cues. *Bulletin of the Psychonomic Society*, 9(5):353–356. (p. 14.)

Damasio, A. (1994). *Descartes' Error: Emotion, Reason, and the Human Brain*. Quill. (p. 1.)

Darwin, C. (1872). *The Expression of the Emotions in Man and Animals*. John Murray, London. (p. 1 and 13.)

Davis, R. B., Ounpuu, S., Tyburski, D., and Gage, J. R. (1991). A gait analysis data collection and reduction technique. *Human Movement Science*, 10(5):575–587. (p. 60.)

de Gelder, B., Snyder, J., Greve, D., Gerard, G., and Hadjikhani, N. (2004). Fear fosters flight: A mechanism for fear contagion when perceiving emotion expressed by a whole body. *Proceedings of the National Academy of Sciences of the United States of America*, 101(47):16701–16706. (p. 1.)

de Leva, P. (1996). Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters. *Journal of Biomechanics*, 29 (9):1223–30. (p. 2, 57, 60, 64, 65, and 66.)

Delp, S., Loan, J., Hoy, M., Zajac, F. E., Topp, E., and Rosen, J. (1990). An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *Biomedical Engineering, IEEE Transactions on*, 37(8):757–767. (p. 1.)

Dumas, R., Chèze, L., and Verriest, J.-P. (2007). Adjustments to mcconville et al. and young et al. body segment inertial parameters. *Journal of Biomechanics*, 40(3):543 – 553. (p. 2, 57, 60, and 64.)

Ekman, P. (1992). An argument for basic emotions. *Cognition & Emotion*, 6(3-4):169–200. (p. 13.)

Ekman, P. and Friesen, W. (1978). *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, Palo Alto. (p. 1 and 13.)

Ekman, P. and Friesen, W. V. (1971). Constants across cultures in the face and emotion. *Journal of Personality and Social Psychology*, 17(2):124–129. (p. 1.)

Featherstone, R. (1983). The calculation of robot dynamics using articulated-body inertias. *The International Journal of Robotics Research*, 2(1):13–30. (p. 43.)

Featherstone, R. (2005). Efficient factorization of the joint-space inertia matrix for branched kinematic trees. *The International Journal of Robotics Research*, 24(6):487–500. (p. 47.)

Featherstone, R. (2006). Plucker basis vectors. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pages 1892–1897. (p. 33.)

Featherstone, R. (2008). *Rigid Body Dynamics Algorithms*. Springer. (p. 29, 33, 36, 39, 40, 43, 47, 48, 50, and 51.)

Featherstone, R. (2010a). A beginner's guide to 6-d vectors (part 1). *Robotics Automation Magazine, IEEE*, 17(3):83–94. (p. 33.)

Featherstone, R. (2010b). A beginner's guide to 6-d vectors (part 2). *Robotics Automation Magazine, IEEE*, 17(4):88–99. (p. 33.)

Gendron, M. and Feldman Barrett, L. (2009). Reconstructing the past: A century of ideas about emotion in psychology. *Emotion Review*, 1(4):316–339. (p. 13.)

Geyer, H., Seyfarth, A., and Blickhan, R. (2006). Compliant leg behaviour explains basic dynamics of walking and running. *Proceedings of the Royal Society B: Biological Sciences*, 273(1603):2861–2867. (p. 1.)

Gill, P., Murray, W., and Wright, M. (1981). *Practical optimization*. Academic Press. (p. 83.)

Gill, P. E., Murray, W., and Saunders, M. A. (1995). User's guide for qpopt 1.0: A fortran package for quadratic programming. (p. 84.)

Guennebaud, G., Jacob, B., et al. (2010). Eigen v3. http://eigen.tuxfamily.org. (p. 33.)

Hamner, S. R., Seth, A., and Delp, S. L. (2010). Muscle contributions to propulsion and support during running. *Journal of Biomechanics*, 43(14):2709 – 2716. (p. 1.)

Hartl, R. F., Sethi, S. P., and Vickson, R. G. (1995). A survey of the maximum principles for optimal control problems with state constraints. *SIAM Review*, 37(2):pp. 181–218. (p. 79.)

Hatz, K. (2014). *Efficient Numerical Methods for Hierarchical Dynamic Optimization with Application to Cerebral Palsy Gait Modeling*. PhD thesis, University of Heidelberg. (p. 3.)

Hicheur, H., Kadone, H., Grèzes, J., and Berthoz, A. (2013a). The combined role of motion-related cues and upper body posture for the expression of emotions during human walking. In Mombaur, K. and Berns, K., editors, *Modeling, Simulation and Optimization of Bipedal Walking*, volume 18 of *Cognitive Systems Monographs*, pages 71–85. Springer Berlin Heidelberg. (p. 1 and 15.)

Hicheur, H., Kadone, H., Grèzes, J., and Berthoz, A. (2013b). Perception of emotional gaits using avatar animation of real and artificially synthesized gaits. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 460–466. (p. 15 and 16.)

Hicheur, H., Terekhov, A. V., and Berthoz, A. (2006). Intersegmental coordination during human locomotion: Does planar covariation of elevation angles reflect central constraints? *Journal of Neurophysiology*, 96(3):1406–1419. (p. 13 and 108.)

Hjortsjö, C. (1969). *Man's Face and Mimic Language*. Studen litteratur. (p. 1.)

Horn, B. K. P. (1987). Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642. (p. 60.)

Huis in 't Veld, E. M., van Boxtel, G. J., and de Gelder, B. (2014a). The body action coding system ii: Muscle activations during the perception and expression of emotion. *Frontiers in Behavioral Neuroscience*, 8(330). (p. 1 and 16.)

Huis in 't Veld, E. M. J., Van Boxtel, G. J. M., and de Gelder, B. (2014). The body action coding system i: Muscle activations during the perception and expression of emotion. *Social Neuroscience*, 9(3):249–264. PMID: 24564288. (p. 1 and 16.)

Ierusalimschy, R., Figueiredo, L. H. d., and Celes, W. (2006). *Lua 5.1 Reference Manual*. Lua.Org. (p. 64.)

Ivanenko, Y. P., d'Avella, A., Poppele, R. E., and Lacquaniti, F. (2008). On the origin of planar covariation of elevation angles during human locomotion. *Journal of Neurophysiology*, 99(4):1890–1898. (p. 13 and 108.)

Jain, A. (1991). Unified formulation of dynamics for serial rigid multibody systems. *Journal of Guidance, Control, and Dynamics*, 14(3):531–542. (p. 43.)

Jain, A. (2011). *Robot and Multibody Dynamics*. Springer US. (p. 29.)

James, W. (1884). What is an emotion? *Mind*, (34):188–205. (p. 1 and 13.)

Johansson, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, 14(2):201–211. (p. 14.)

Khalil, W. and Dombre, E. (2004). *Modeling, Identification and Control of Robots*. Kogan Page Science paper edition. Elsevier Science. (p. 29.)

Kirk, A. G., O'Brien, J. F., and Forsyth, D. A. (2005). Skeletal parameter estimation from optical motion capture data. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*, pages 782–788. (p. 60.)

Koch, K. H. (2015). *Using Model-based Optimal Control for Conceptional Motion Generation for the Humannoid Robot HRP-2 14 and Design Investigations for Exo-Skeletons*. PhD thesis, Heidelberg University. (p. 52.)

Kokkevis, E. and Metaxas, D. (1998). Efficient dynamic constraints for animating articulated figures. *Multibody System Dynamics*, 2(2):89–114. (p. 47.)

Koschorreck, J. and Mombaur, K. (2009). Optimization of sommersaults and twists in platform diving. *Computer Methods in Biomechanics and Biomedical Engineering*, 12(2-1). (p. 2.)

Krüger, S., Sokolov, A. N., Enck, P., Krägeloh-Mann, I., and Pavlova, M. A. (2013). Emotion through locomotion: Gender impact. *PLoS ONE*, 8(11):e81716. (p. 14.)

Kuo, A. D. (2007). The six determinants of gait and the inverted pendulum analogy: A dynamic walking perspective. *Human Movement Science*, 26(4):617 – 656. European Workshop on Movement Science 2007, European Workshop on Movement Science 2007. (p. 1.)

Lee, S. J. and Hidler, J. (2008). Biomechanics of overground vs. treadmill walking in healthy individuals. *Journal of Applied Physiology*, 104(3):747–755. (p. 113.)

Leineweber, D. (1995). Analyse und Restrukturierung eines Verfahrens zur direkten Lösung von Optimal-Steuerungsproblemen. Master's thesis, Ruprecht-Karls-Universität Heidelberg. (p. 79.)

Leineweber, D., Bauer, I., Schäfer, A., Bock, H., and Schlöder, J. (2003). An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization (Parts I and II). *Computers & Chemical Engineering*, 27:157–174. (p. 80.)

Maplesoft (2014). Maple 18. online. (p. 33.)

Masani, K., Kouzaki, M., and Fukunaga, T. (2002). Variability of ground reaction forces during treadmill walking. *Journal of Applied Physiology*, 92(5):1885–1890. (p. 113.)

Mather, G. and Murdoch, L. (1994). Gender discrimination in biological motion displays based on dynamic cues. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 258(1353):273–279. (p. 14.)

Matthew Millard, Thomas Uchida, A. S. and Delp, S. L. (2013). Flexing computational muscle: Modeling and simulation of musculotendon dynamics. *Journal of Biomechanical Engineering*, 135. (p. 72.)

Mombaur, K. (2009, published online June 2008). Using optimization to create self-stable human-like running. *Robotica*, 27:321–330. (p. 2.)

Mombaur, K., Truong, A., and Laumond, J.-P. (2010). From human to humanoid locomotion – an inverse optimal control approach. *Autonomous Robots*, 28(3):369–383. (p. 3.)

Motion Lab Systems (2008). The c3d file format user guide. Technical report, Motion Lab Systems. (p. 18 and 71.)

Nakamura, Y. and Hanafusa, H. (1986). Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement, and Control*, 108(3):163–171. (p. 71.)

Nakamura, Y., Yamane, K., Fujita, Y., and Suzuki, I. (2005). Somatosensory computation for man-machine interface from motion-capture data and musculoskeletal human model. *Robotics, IEEE Transactions on*, 21(1):58–66. (p. 1 and 72.)

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization.* Springer Science+Business Media, LLC. (p. 46 and 83.)

O'Brien, J. F., Bodenheimer, R. E., Brostow, G. J., and Hodgins, J. K. (2000). Automatic joint parameter estimation from magnetic motion capture data. In *Proceedings of Graphics Interface 2000*, pages 53–60. (p. 60.)

Omlor, L. and Giese, M. (2006). Blind source separation for over-determined delayed mixtures. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*, pages 1049–1056. MIT Press, Cambridge, MA. (p. 15.)

Omlor, L. and Giese, M. A. (2007). Extraction of spatio-temporal primitives of emotional body expressions. *Neurocomput.*, 70(10-12):1938–1942. (p. 1 and 15.)

Orin, D. and Goswami, A. (2008). Centroidal momentum matrix of a humanoid robot: Structure and properties. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 653–659. (p. 46.)

Perry, J. and Burnfield, J. M. (2010). *Gait Analysis: Normal and Pathological Function.* SLACK Incorporated. (p. 11.)

Plutchik, R. (1980). *Theories of emotion.* Academic Press, New York. (p. 13.)

Pontryagin, L. S. (1987). *Mathematical theory of optimal processes.* CRC Press. (p. 79.)

Potschka, A., Bock, H. G., and Schlöder, J. P. (2009). A minima tracking variant of semi-infinite programming for the treatment of path constraints within direct solution of optimal control problems. *Optimization Methods and Software*, 24(2):237–252. (p. 81.)

Pozzo, T., Berthoz, A., and Lefort, L. (1990). Head stabilization during various locomotor tasks in humans. *Experimental Brain Research*, 82(1):97–106. (p. 13.)

Roether, C. L., Omlor, L., Christensen, A., and Giese, M. A. (2009). Critical features for the perception of emotion from gait. *Journal of Vision*, 9(6). (p. 1, 15, and 17.)

Sartori, M., Reggiani, M., Farina, D., and Lloyd, D. G. (2012). Emg-driven forward-dynamic estimation of muscle force and joint moment about multiple degrees of freedom in the human lower extremity. *PLoS ONE*, 7(12):e52618. (p. 112.)

Schultz, G. and Mombaur, K. (2010). Modeling and optimal control of human-like running. *Mechatronics, IEEE/ASME Transactions on*, 15(5):783 –792. (p. 2.)

Schwartz, M. H. and Rozumalski, A. (2005). A new method for estimating joint parameters from motion data. *Journal of Biomechanics*, 38(1):107 – 116. (p. 59.)

Seyfarth, A., Geyer, H., Günther, M., and Blickhan, R. (2002). A movement criterion for running. *Journal of Biomechanics*, 35(5):649 – 655. (p. 1.)

Shabana, A. (2005). *Dynamics of Multibody Systems.* Cambridge University Press. (p. 29.)

Shoemake, K. (1985). Animating rotation with quaternion curves. *SIGGRAPH Comput. Graph.*, 19(3):245–254. (p. 39.)

Shreiner, D. and Group, T. K. O. A. W. (2009). *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1.* Addison-Wesley Professional, 7th edition. (p. 71.)

Stasse, O., Escande, A., Mansard, N., Miossec, S., Evrard, P., and et Kheddar, A. (2008). Real-time (self)-collision avoidance task on a HRP-2 humanoid robot. In *icra08*, Las Passadenas, USA. (p. 72.)

Strack, F., Martin, L. L., and Stepper, S. (1988). Inhibiting and facilitating conditions of the human smile: A nonobtrusive test of the facial feedback hypothesis. *Journal of Personality and Social Psychology*, 54(5):768–777. (p. 1.)

Sugihara, T. (2009). Solvability-unconcerned inverse kinematics based on Levenberg-Marquardt method with robust damping. In *9th IEEE-RAS International Conference on Humanoid Robots December 7-10*, volume 27, pages 984–991. (p. 68 and 71.)

Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(4):376–380. (p. 59.)

Venture, G., Kadone, H., Zhang, T., Grèzes, J., Berthoz, A., and Hicheur, H. (2014). Recognizing emotions conveyed by human gait. *International Journal of Social Robotics*, 6(4):621–632. (p. 15.)

von Stryk, O. (2000). User's guide for DIRCOL (version 2.1): A direct collocation method for the numerical solution of optimal control problems. Technical report, Fachgebiet Simulation und Systemoptimierung (SIM), Technische Universität Darmstadt (2000, Version of November 1999). (p. 79.)

von Stryk, O. and Bulirsch, R. (1992). Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, 37(1):357–373. (p. 79.)

Wallbott, H. G. (1998). Bodily expression of emotion. *European Journal of Social Psychology*, 28(6):879–896. (p. 15, 17, and 106.)

Winter, D. A. (2009). *Biomechanics and Motor Control of Human Movement.* John Wiley & Sons, Inc., Hoboken, New Jersey, fourth edition edition. (p. 24.)