Universidade do Minho
Escola de Engenharia

Philippe Daniel Pinto Ferreira

Development of a two-dimensional biomechanical multibody model for the analysis of the human gait with an ankle-foot orthosis

October 2012

Development of a two-dimensional biomechanical multibody model for the analysis of the human gait with an ankle-foot orthosis

Philippe Daniel Pinto Ferreira

UMinho | 2012

Universidade do Minho
Escola de Engenharia

Philippe Daniel Pinto Ferreira

Development of a two-dimensional biomechanical multibody model for the analysis of the human gait with an ankle-foot orthosis

M.Sc. Dissertation
Integrated Master in Biomedical Engineering
Biomaterials, Biomechanics and Rehabilitation

Dissertation done under the supervision of
João Paulo Flores Fernandes

October 2012

DECLARAÇÃO

Nome: Philippe Daniel Pinto Ferreira

Título dissertação: Development of a two- dimensional biomechanical multibody model for the analysis of the human gait with an ankle-foot orthosis

Orientador: João Paulo Flores Fernandes Ano de conclusão: 2012

Designação do Mestrado: Mestrado Integrado em Engenharia Biomédica

Universidade do Minho, ___/___/_____

Assinatura: _____

The total number of persons with paralysis, deformity or orthopedic impairments that use orthoses is expected to reach 7.3 million by the year 2020.

American Academy of Orthotists and Prosthetists

ACKNOWLEDGMENTS

This dissertation wouldn't have been possible without the help and support of the kind people around me.

First and foremost, I would like to thank my family who encourages and stimulates me during the whole year.

I am sincerely and heartily grateful to my supervisor, Professor Paulo Flores, for believing in my potential and accepting to supervise my project. His insights and periodic advices were essential to complete this work.

I am deeply grateful to the whole team of the Department for Mechanics and Robotics from the Duisburg-Essen University. I was very well welcomed and learnt a lot during my semester there. I would like to thank in particular to Professor Andrés Kecskeméthy who was kind to receive me in his research group. He was patient enough to clarify all my doubts and his motivation, interest and support help me to overcome all the obstacles.

I would also like to thank Professor Mario Siebler from the Heinrich-Hein University, in Düsseldorf for receiving me in his clinic, giving me the clinical perspective to the ankle foot orthoses, providing me two pairs of orthoses and for his availability to find patients for data acquisition in the gait lab.

This project was developed under the project DACHOR and I would like to thank the MIT Portugal Program for this opportunity to work in this interesting topic and for the fellowship I was awarded.

Finally, I would like to thanks all my friends, in Portugal, Germany and elsewhere, who gave me constant moral support and boost my productivity.

TITLE

Development of a two-dimensional biomechanical multibody model for the analysis of the human gait with an ankle-foot orthosis

KEYWORDS

Biomechanics

Ankle-foot orthosis

Gait analysis

Human body simulation

Forward dynamics

ABSTRACT

Ankle-foot orthoses are orthotic devices that support the ankle joint and are appropriate for several pathologies, mostly the ones that cause dropfoot, which is caused by an ankle joint deficiency.

In the present work, a planar multibody model of the human body in the sagittal plane was developed. For this purpose, the MOBILE computational program was utilized. The model simulates the lower limbs and is made of 9 rigid bodies. It has 12 DOFs and is prepared for reproducing kinematic data acquired in a gait lab.

Kinematic measurements were obtained in a gait lab from a healthy subject, with and without plastic ankle foot orthoses worn on both feet. The results obtained showed that with the orthoses, the ankle joint behavior is similar to a linear torsional spring, with almost no hysteresis.

Ankle kinematics, measured in the gait lab with and without orthoses, were successfully reproduced by forward dynamics using the multibody model developed, which allows for the validation of the presented approach.

Furthermore, it was concluded that ankle foot orthoses can be modeled as a spring element acting at the ankle joint, and the use of an ankle foot orthosis reduces the muscle activation at the ankle in about 15%.

TITEL

Entwicklung eines Zwei-Dimensional Biomechanische MB Modell für die Analyse der menschlichen Gang mit einem Knöchel-Fuß Orthese

STICHWORTE

Biomechanik

Knöchel-Fuss Orthesen

Ganganalyse

Menschliche Körper Simulation

Vorwärts Dynamik

ABSTRAKT

Knöchel-Fuß-Orthesen sind orthopädische Geräte, die das Sprunggelenk zu unterstützen und sind für verschiedene Erkrankungen, vor allem diejenigen, die dropfoot, die von einer Sprunggelenks-Mangel verursacht wird dazu führen, angemessen.

In der vorliegenden Arbeit wurde ein planarer Multibody Modell des menschlichen Körpers in der Sagittalebene entwickelt. Zu diesem Zweck wurde das MOBILE Rechenprogramm verwendet. Das Modell simuliert die unteren Extremitäten und wird von 9 starren Körpern. Es verfügt über 12 Freiheitsgrade und ist für die Wiedergabe kinematische Daten in einer Ganglabors erworbenen vorbereitet.

Kinematische Messungen wurden in einem Ganglabors von einer gesunden Person erhalten, mit und ohne Kunststoff Sprunggelenk Orthesen an beiden Füßen getragen. Die erhaltenen Ergebnisse zeigten, dass mit den Orthesen, das Sprunggelenk Verhalten ähnlich einer linearen Torsionsfeder ist, fast ohne Hysterese.

Knöchel Kinematik, in der Ganglabors mit und ohne Orthesen gemessen wurden erfolgreich von Vorwärtsdynamikanalyse Verwendung der Multibody Modell entwickelt, das für die Validierung der dargebotenen Ansatz ermöglicht reproduziert.

Darüber hinaus wurde festgestellt, dass Sprunggelenk Orthesen als Feder wirkende Element am Sprunggelenk modelliert werden können, und die Verwendung eines Fußheberorthese verringert die Muskelaktivität am Knöchel in etwa 15%.

TÍTULO

Desenvolvimento de um modelo biomecânico multibody bidimensional para a análise da marcha humana com uma ortótese do tornozelo

PALAVRAS CHAVE

Biomecânica

Ortótese do tornozelo

Análise da marcha humana

Simulação do corpo humano

Dinâmica direta

RESUMO

As ortóteses do tornozelo são dispositivos ortopédicos que apoiam a articulação do tornozelo e são indicados para uma variedade de patologias, nomeadamente as que causam pé pendente, que é uma deficiência na mobilidade do tornozelo.

No presente trabalho, um modelo multibody planar do corpo humano no plano sagital foi criado. Para tal, o *software* MOBILE foi usado. O modelo simula os membros inferiores e é composto por nove corpos rígidos. Possui 12 graus de liberdade e está preparado para usar dados cinemáticos adquiridos num laboratório de análise da marcha humana como restrições de guiamento.

Dados cinemáticos foram obtidos num laboratório de análise da marcha humana, a partir de um indivíduo saudável, com e sem ortóteses plásticas em ambos os pés. Os resultados mostraram que, com a ortótese, o comportamento da articulação do tornozelo é semelhante a uma mola de torção linear, praticamente sem histerese.

Os dados cinemáticos do tornozelo, medidos no laboratório de marcha, com e sem ortótese, foram reproduzidos com sucesso por uma dinâmica direta, utilizando o modelo multibody desenvolvido, o que validou a abordagem utilizada.

Todas as metodologias encontram-se descritas e explicadas nesta tese e concluiu-se que a ortóteses do tornozelo podem ser modeladas como uma mola de torsão que actua na articulação do tornozelo. Concluiu-se também que a utilização de uma ortótese do tornozelo por uma pessoa saudável reduz a activação muscular do mesmo em cerca de 15%.

TABLE OF CONTENTS

ABBREVIATIONS AND ACRONYMS

2D	*Two-dimensional*
3D	*Three-dimensional*
3PP	*Three point pressure*
AAFO	*Active ankle-foot orthosis*
AFO	*Ankle-foot orthosis*
CM	*Center of mass*
DOF	*Degree-of-freedom*
FEM	*Finite element model*
GRF	*Ground reaction force*
HAT	*Head, arms and trunk*
HKAFO	*Hip-knee-ankle-foot orthosis*
HS	*Heel strike*
IR	*Infra-red*
KAFO	*Knee-ankle-foot orthosis*
LFEO	*Left Femur Origin*
LHEE	*Left Heel*
LHJC	*Left Hip Joint Center*
LHUP	*Left Posterior Humerus*
LTIO	*Left Tibia Origin*
LTOE	*Left Toe*
MBS	*Multibody system*
MSE	*Mean squared error*
***o*HS**	*Opposite heel strike*
***o*TO**	*Opposite toe off*
PELO	*Pelvis Origin*
PLS	*Posterior leaf spring*
PPT	*Pain pressure threshold*
PP	*Polypropylene*

RFEO	*Right Femur Origin*
RHEE	*Right Heel*
RHJC	*Right Hip Joint Center*
RHUP	*Right Posterior Humerus*
RTIO	*Right Tibia Origin*
RTOE	*Right Toe*
RTOETIP	*Right Toe Tip*
SIMM	*Software for Interactive Musculoskeletal Modeling*
THKAFO	*Trunk-hip-knee-ankle-foot orthosis*
TO	*Toe off*

LIST OF FIGURES

Chapter 1

Chapter 2

Chapter 3

Chapter 4

LIST OF TABLES

Chapter 1

Chapter 2

Chapter 3

Chapter 4

CHAPTER 1

INTRODUCTION

1.1 Motivation

Biomechanics is the scientific domain which deals with the study of biological systems, such as the human body, using physical concepts and mechanical engineering methodologies (Whittle, 2007). Biomechanics is a large area and it is possible to find biomechanics engineers studying many different aspects of biological systems, such as rehabilitation, ergonomics, biomaterials and biotribology, biofluid, among others.

The human gait, in particular, is easily identified as a mechanical process and since it is performed by a biological system, it is appropriate to study it as a mechanical system. In turn, mechanical engineering is a vast subject that involves Newtonian mechanics and materials sciences but the most relevant concepts for the gait analysis are time, mass, force, center of gravity, torques, and motion, both linear and angular (Whittle, 2007).

Since biological systems are commonly more complex than man-built mechanical systems, the biomechanical approach employed to for their study can be extremely mathematical and numerical methods are often applied. However the basic principles are easy to understand.

Biomechanics research is done in an iterative process of hypothesis and verification that normally include modeling, computer simulation and experimental validation of the results.

The present work focuses a well-known medical device, the ankle-foot orthosis (AFO). The necessity for orthotic devices is increasing and, since this kind of device plays an important role in restoring the normal gait to patients suffering from ankle disabilities, it is a relevant study subject.

Lower limb orthoses were the most common orthotic device used, covering 55% of the orthoses provided in the United States of America (USA) in 2007. AFOs in particular represent almost half of this category with 26% and proved to be the most common type of orthoses (American Board for Certification in Orthotics Prosthetics and Pedorthics Inc., 2007). Nowadays, the elderly population is increasing and this growth in the number of elderly people will have an impact in the number of AFOs needed. This fact will have economic implications and AFOs proved to be an increasing market, since their base cost was between $500 and $700, according to a Medicare review payment data for the years 2001 to 2006 (American Orthotic and Prosthetic Association Inc., 2008).

1.2 Objectives and thesis organization

The main goal of this work is to develop a biomechanical multibody (MBS) model for human gait analysis that includes the AFO properties. The model must be able to simulate the orthosis effect on a patient using an AFO. Another purpose of this work deals with the effect of AFOs on the energy consumption associated to the human gait.

The specific objectives of this investigation can be listed as follows:

- To build up a planar model of the lower limbs in the sagittal plane using the subject-oriented biomechanics library MOBILE;
- To acquire kinematic data on a gait lab (with and without orthosis);
- To use the kinematic data as an input to drive the MBS model;
- To reproduce the results using forward dynamics;
- To include the AFOs properties in the model;
- To compare the energy consumption with and without AFOs.

The model here proposed is a planar model that uses the subject-oriented biomechanics library MOBILE. During the gait cycle, the major forces and the major amplitude of movements are expected in the sagittal plane (Silva et al., 2010). For this reason and for simplicity purposes, the developed model is a 2D approach in the sagittal plane. Since the AFO is a passive device, it does not have an actuator and the model cannot have a power supply.

This dissertation is organized as follows:

In Chapter 1, an overview of the dissertation is provided, objectives are defined and a literature review is made to characterize the existing MBS models of the human body that includes AFOs.

In Chapter 2, the anatomical and biomechanical perspective of the ankle-foot complex is presented. Motion allowed by the ankle joint is defined and the phases of the gait cycle are described. All kinds of lower limb orthoses are covered with particular attention to AFOs. Pathologies leading to the AFO necessity are listed and the biomechanical effect of these orthotic devices is explained.

Chapter 3 describes the MBS model developed in this work. The software used, MOBILE, is presented and the first models developed in this programming environment are referred. The MBS model is described, with particular emphasis on the foot and on

the contact foot model. A short literature review on contact foot models is also done to contextualize the developed model.

Chapter 4 covers the methodologies employed. First, the data acquisition in the gait lab is referred and explained. Then, the problems faced and their corresponding solutions are presented. Ankle moment over ankle angle and ankle angle over time plots are studied, since they represent the incoming results. The model was tested with barefoot kinematical data which were reproduced successfully by forward dynamics. The same methodologies were applied in order to reproduce the kinematical data measured in the gait lab with an orthosis, but the ankle moment was reduced and the AFO was replaced by a spring-damper element applied at the ankle joint.

Finally, Chapter 5 summarizes the main conclusions of the present work, as well as the advantages and limitations of the methodologies employed. Some suggestions for future developments are also presented.

1.3 Literature review

Much research has been made attending to understand all the process of walking, and nowadays, it is a well-studied area (Abboud, 2002, Whittle, 2007). The human walking is a process that involves many organs, such as the brain, spinal cord, peripheral nerves, muscles, bones and joints, and to understand it correctly it is crucial to know the basics about anatomy, physiology and biomechanics (Whittle, 2007).

Some work has been done on modeling the normal and the pathological gait, regarding to the muscle activation (Lamontagne et al., 2002) and trying to understand pain (Callaghan and Baltzopoulos, 1994). Many research work have been published on the characterization of the mechanical properties of AFOs (Yamamoto et al., 1993, Bregman et al., 2009, Crabtree and Higginson, 2009, Lai et al., 2010) or the effect of this type of orthosis on the pathological gait (Romkes and Brunner, 2002, Gordon et al., 2006, Brehm et al., 2008). However, there are not many available literature on the modeling of the human gait with an AFO.

The first attempt to model an AFO with the ankle-foot complex was published by Chu et al. (1995). Before this date, only few papers related to the behavior of AFOs under static forces can be found (Leone et al., 1988, Leone et al., 1991). These authors compare results from beam theory, finite elements and experimental results in order to predict the AFO deflection.

In 1995, a 3D finite element model (FEM) of an AFO together with the entire ankle-foot complex was developed using ADINA software (see Fig. 1.1)(Chu et al., 1995). A static analysis was performed for two of the most relevant instants in gait cycle, namely the heel strike and the toe off. The authors showed that significant stress concentrations occur in the heel and the neck regions of the AFO. The peak tensile stress (0.8 MPa) and compressive stress (1.6 MPa) occurred in the neck region of the orthosis during toe off and in the heel region during the heel strike, respectively. Since then, Chu's group has published other relevant works about stress analysis of polypropylene AFOs (Chu and Reddy, 1995, Chu and Feng, 1998, Chu, 2000, Chu, 2001).

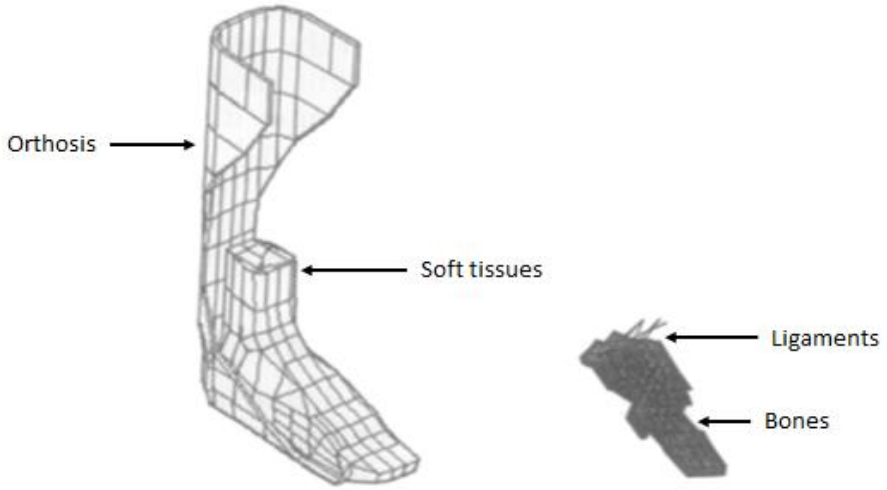


Figure 1.1 3D FEM of the ankle-foot orthosis system (Chu et al., 1995)

Most recently, in 2008, a new model for the gait simulation combining the AFO function and steppage gait was developed by Jamshidi et al. (2008). In this study, a five-segment MBS model of the human body in the sagittal plane was created, as it is illustrated in Fig. 1.2a. This AFO considers the role of a spring-damper system between the foot and the shank (see Fig. 1.2b) that compensate muscle weakness. The dynamic model was solved by inverse dynamic, in which the value of the spring constant (k) varies from 1 to 75 N.m.rad^{-1}, and the damper constant (c) is null. Comparing the value of the calculated torque in ankle joints with the value for normal gait found in the literature, it was concluded that the ideal values were $k = 1\ N.m/rad$ and $c = 0\ N.s/m$.

Later on, the same group (Jamshidi et al., 2009) applied their previous model, using inverse dynamics, with kinematics data from a Guillan-Barré patient with

dropfoot disorder. The aim of the study was to estimate how the spring coefficient affects the torque generated at the ankle to optimize the human gait.

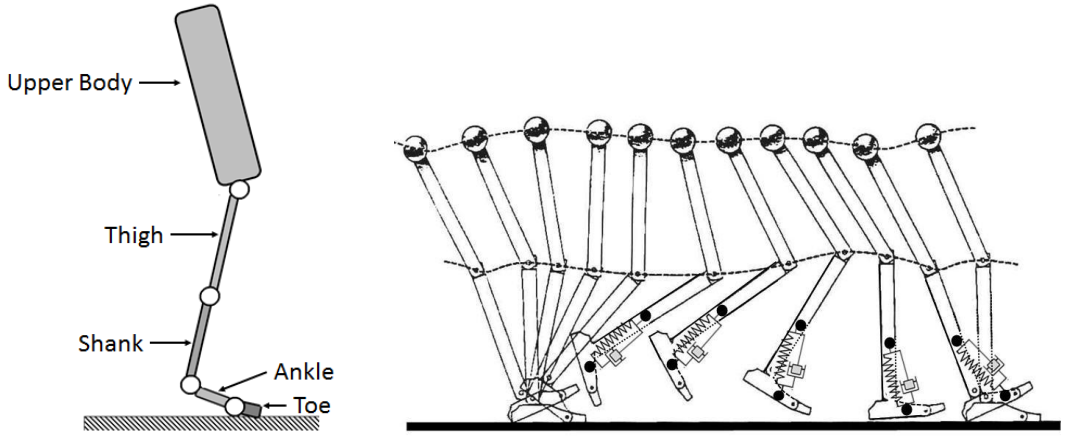
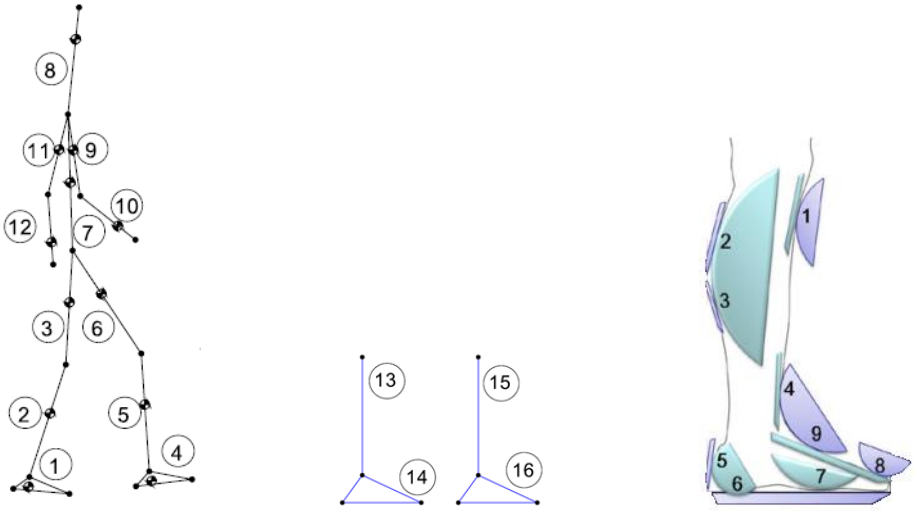


Figure 1.2 (a) Five-segment diagram of human's body and **(b)** schematic of gait cycle for quantifying the function of AFO (Jamshidi et al., 2008)

Crabtree and Higginson (2009) developed an AFO model as an adjustable torque applied at the ankle joint. This torque varies in function of joint angle and velocity and its behavior was obtained experimentally with a dynamometer. In this paper, the authors apply this model on the right leg of a 9 degrees-of-freedom (DOF), 11 segment model of the musculoskeletal system developed in SIMM (Software for Interactive Musculoskeletal Modeling, Musculographics, Inc.) with 15 muscles per leg. The right ankle presented passive stiffness and the model was solved in forward dynamics. Some changes in neuromuscular control of ankle plantarflexors and dorsiflexors were observed in response to the moments provided by the AFOs.

Silva et al. (2010) developed a new MBS model of an active AFO that can be integrated in a whole-body MBS human model with 12 DOFs, as it is shown in Fig. 1.3a. This model (see Fig. 1.3b) was developed in the MATLAB environment and it was created to evaluate the contact forces generate in the lower limb/AFO interface. The aim of this study was to evaluate the comfort of the patient, checking if the contact forces were below the pain pressure thresholds (PPTs). The contact model used is a contact model between a deformable sphere and a rigid plane (see Fig. 1.3c). According to this investigation, there are nine contact points between the orthosis and the lower limb, as it is presented in Tab. 1.1. Kinematic data as well as ground reaction forces (GRFs) of a healthy subject were acquired in a gait lab. Two simulations were performed, in forwards dynamics, using the biomechanical model (see Fig. 1.3a) and driving the joints with the kinematic data acquired. The first simulation used all the

kinematic data while in the second one, the ankle joint was considered to be free in order to simulate the dropfoot as a total absence of muscular action. The second simulation included the AFO model (see Fig. 1.3b) in both legs. The AFOs prescribed the kinematic data of the ankle joint. The weight of each rigid segment and the weight of the AFOs were applied to the center of mass of each segment and on the ankle joint of the AFOs, respectively. Finally, the GRFs were applied under the foot in the first simulation and under the orthoses plantar modules in the second one. The results presented a very close gait pattern. The moments at the ankle joint, knee and hip were calculated and compared, revealing very close values. The pressures at the contact points were also evaluated and it was concluded they were below the PPTs.



(a) **(b)** **(c)**

Figure 1.3 (a) Biomechanical model of the human body constituted by 12 rigid segments interconnected by geometrical ideal revolute joints, **(b)** multibody model of the AFOs and **(c)** schematic locations of sphere/plane contact points between the AFO and the lower limb (Silva et al., 2010)

Table 1.1 Description of the materials in the contact points (Silva et al., 2010)

Contact Point	Deformable Sphere	Rigid Plane
1	Straps	Bone prominences
2	Straps	Bone prominences
3	Straps	Bone prominences
4	Soft tissues	Structural nondeforming orthosis
5	Soft tissues	Structural nondeforming orthosis
6	Soft tissues	Structural nondeforming orthosis
7	Soft tissues	Structural nondeforming orthosis
8	Soft tissues	Structural nondeforming orthosis
9	Straps	Bone prominences

Most recently, Bregman et al. (2011) developed a forward-dynamic conceptual walking 2D model with a passive spring at the ankle representing the AFO. This model

(see Fig. 1.4) consists of 7 rigid segments powered by two constant torques applied at the hip. Those torques are consistent with compensation strategies in patients with impaired push-off. All the joints were frictionless. The knee joints were locked in full extension by jambs during stance phase and modeled as hinge joints with hyperextension stop during swing phase. During swing and stance phase, a flexion torque and an extension torque were applied on the upper leg, respectively. The interactions between the model and the floor were modeled as unilateral constraints located at the heel and toes. These constraints were activated based on kinematic events, that is when the heel or the toes touched the floor and deactivated when the vertical forces between the contact points of the foot and the floor decreased to zero. The ankle was modeled as a free hinged joint, showing a completely paralyzed ankle, and the only moment acting at this articulation was the linear rotational spring. Simulations were performed in MATLAB by solving the equations of motion in a forward dynamic fashion for a broad range of AFO stiffnesses. Walking speed and step length were kept constant by adapting the magnitude of the hip flexion and extension torques. The results were computed for seven different velocities. The energy cost of walking was calculated as the sum of the amount of positive work done by the two constant hip torques. Finally, it was concluded that the patient energy cost of walking is directly dependent on the AFO stiffness and the proper choice of orthosis reduces the energy consumption of walking.

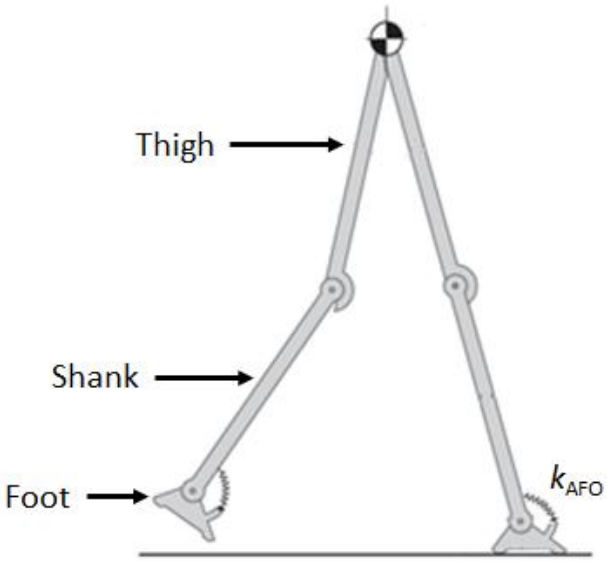


Figure 1.4 Representation of the 2D, 7-segments model with representation of the AFO as a passive spring (Bregman et al., 2011)

1.4 Contributions of the thesis

The first three chapters of the thesis consist of a literature review and important background knowledge is collected. A literature review on the modeling of the human gait with an AFO is presented. The ankle-foot complex anatomy, gait cycle and lower limb orthoses are characterized with particular emphasis on AFOs and their necessity. A few solutions, existing in the literature, on modeling the human body are discussed. The model is described and all stages of its development are listed and reported.

Kinematic measurements are made in a gait lab and it was noticed that when wearing AFOs on both feet, the ankle moment varies linearly with the ankle angle, which proves that the AFO can be modeled as a linear torsional spring applied at the ankle.

The developed model has been tested and proved to be appropriate to reproduce the ankle kinematics during stance phase by forward dynamics. The author has written the whole code which is presented in appendices.

CHAPTER 2

ANKLE AND FOOT ANATOMY, MOTION AND ANKLE-FOOT ORTHOSES

On this chapter, ankle and foot anatomy are described in order to provide basic knowledge of the limbs physiognomy under study. The ankle and foot motion is discussed and a gait cycle analysis is made. Lower limbs orthosis are referred with particular emphasis on the AFOs. Finally, pathologies that may require AFOs are listed and the biomechanics of AFOs are explained.

2.1 Ankle and foot anatomy

The ankle and the foot are a complex mechanical structure consisting of bones, ligaments, tendons and muscles. Ligaments have the function of connecting the bones to each other, reinforcing and stabilizing articulations. Tendons are bands of fibrous tissue connecting the muscles to the bones. Their function is to transfer the force developed by the muscle to the bones, creating movement. The ankle-foot complex plays a primordial role in the human locomotion since it supports the total body weight during the gait cycle.

At the ankle joint, the lower ends of the tibia and fibula form a deep socket which will fits the upper surface of the talus, or ankle bone. The articular surface of the joint is covered with a smooth hyaline cartilage surrounded a synovial membrane filled of synovial fluid. The shape of the bones and the strength of the surrounding ligaments maintain the ankle stable, while still allowing the necessary freedom of movement. A tough fibrous capsule surrounds the joint and is reinforced on each side by the medial (inner) and lateral (outer) ankle ligaments, as Fig. 2.1 shows (Grabner, 2003).

The human foot is made of 26 bones: 7 tarsal bones, 5 metatarsals and 14 phalanges (see Fig. 2.2). The talus articulates with the tibia and the fibula and bears the full weight of the body. It remains above the calcaneus, which is the largest bone foot. It is a large and strong bone whose function is to transmit the body weight from the talus to the ground. The calcaneus has bearing surfaces where it forms joints with the talus above and the cuboid in front. The cuboid has approximately the shape of a cube. The navicular is a relatively small bone located at the talus front and above the cuboid. The three remaining tarsal bones are the cuneiforms, named according to their positions: medial, intermediate and lateral. The medial cuneiform is the largest of these wedge-shaped bones (De Burgh, 2003).

The metatarsals and phalanges are miniature long bones since they consist of a base, a shaft and a head and their function is to provide stability to the foot. The bases

of the metatarsal articulate with the tarsal bones in the midfoot and the heads articulate with each toes' phalanges. There are 14 phalanges in the foot, which are smaller and less mobile than those of the fingers, but their arrangement is the same. Each toe has three phalanges, except for the big toe (or hallux) which has only two (see Fig. 2.2) (De Burgh, 2003).

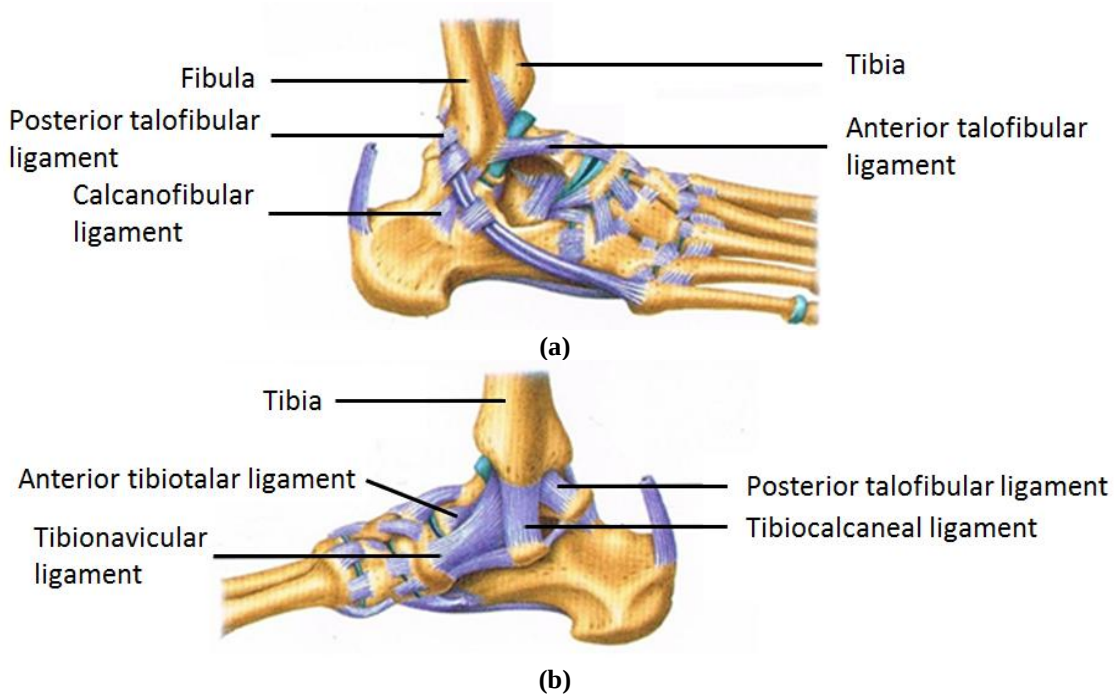


Figure 2.1 Ankle ligaments. **(a)** Lateral/outer and **(b)** Medial/inner ligaments (De Burgh, 2003)

The bones of the foot are arranged in two bridgelike arches that are held in position by ligaments and tendons (see Fig. 2.3).

The foot arches allow the foot to support the total body weight, providing an ideal distribution over soft and hard tissue the foot and leverage while walking. Normally, the forefoot carries about 40% of the weight and the heel carries the remaining 60% (Tortora and Derrickson, 2008).

The two arches are supported by several ligaments that lie on the plantar surface of the foot and provide a strong but flexible base. The three major ligament are the plantar calcaneonavicular ligament, the long plantar ligament and the plantar calcaneocuboid ligament (De Burgh, 2003, Tortora and Derrickson, 2008).

Muscles are soft tissues that contract and exert forces on tendons, which in turn pull one bone (or skin). Most muscles are attached to the articulating bones and are responsible for the articulation movement (Tortora and Derrickson, 2008). There are several muscles responsible for the foot and toes movement. They are located on the leg

and foot and many of them cross the ankle articulation. A series of fibrous bands, *retinacula*, held them firmly in place (De Burgh, 2003). Fig. 2.4 shows the superior and inferior extensor retinaculum whose function is to retain the extensor muscles.

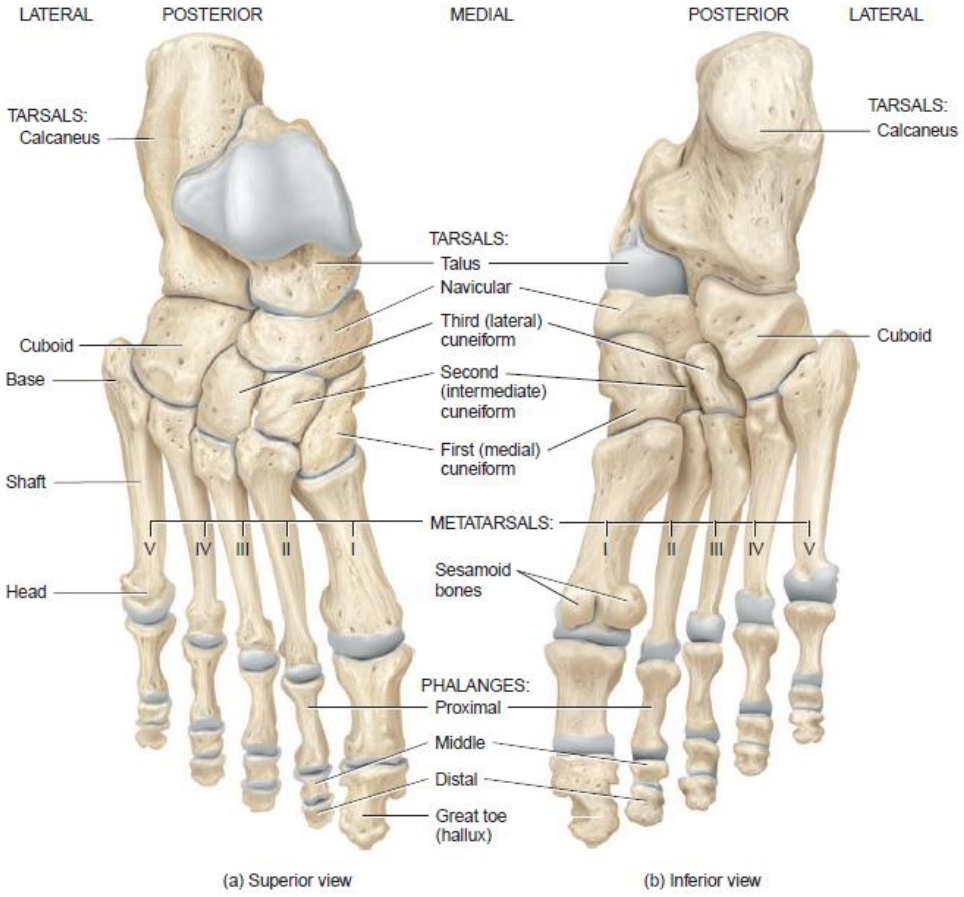


Figure 2.2 Bones of the foot (Tortora and Derrickson, 2008)

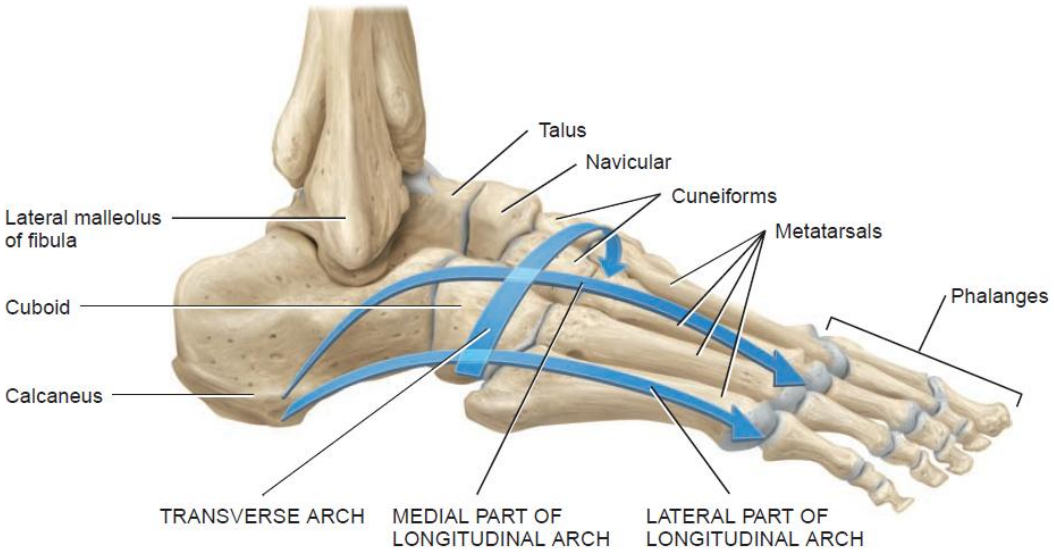


Figure 2.3 Lateral view of arches of the right foot (Tortora and Derrickson, 2008)

Muscles that move the foot and toes can be found on the leg (see Fig. 2.5) and intrinsic muscles of the foot are responsible for the movement of the toes (see Fig. 2.6) (Tortora and Derrickson, 2008).

However, despite the muscles are presented and identified in Figs. 2.5 and 2.6, they will not be described because such details are out of the scope of this thesis.

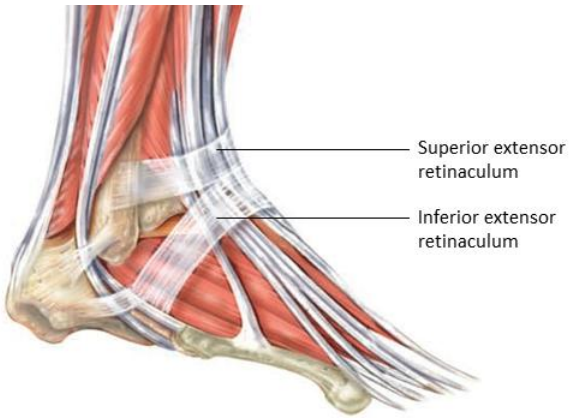


Figure 2.4 Retinacula of the ankle (Tortora and Derrickson, 2008)

2.2 Ankle and foot motion

The ankle-foot complex is a multiarticular mechanical structure consisting of bones, joints and soft tissues. It plays an important role since it is the end part of the lower kinetic chain where external forces are applied (GRFs). The foot is the only part of the body that acts on an external surface, the ground, providing support during standing and stabilizing the body during gait (Abboud, 2002).

The foot has six joints: the ankle, subtalar, midtarsal, tarsometatarsal, metatarsophalangeal and interphalangeal, which are controlled by extrinsic and intrinsic muscles (see Figs. 2.5 and 2.6).

In order to describe the motion of the limbs and joints, three reference planes (see Fig. 2.7) are usually used, namely:

1. A sagittal plane is any plane which divides part of the body into right and left portions. The median plane is the midline sagittal plane, which divides the whole body into right and left halves;
2. The frontal plane, also called coronal plane, divides a body part into anterior and posterior portions;
3. The tranverse plane, or horizontal plane, divides body parts into upper and lower portions (Tortora and Derrickson, 2008).

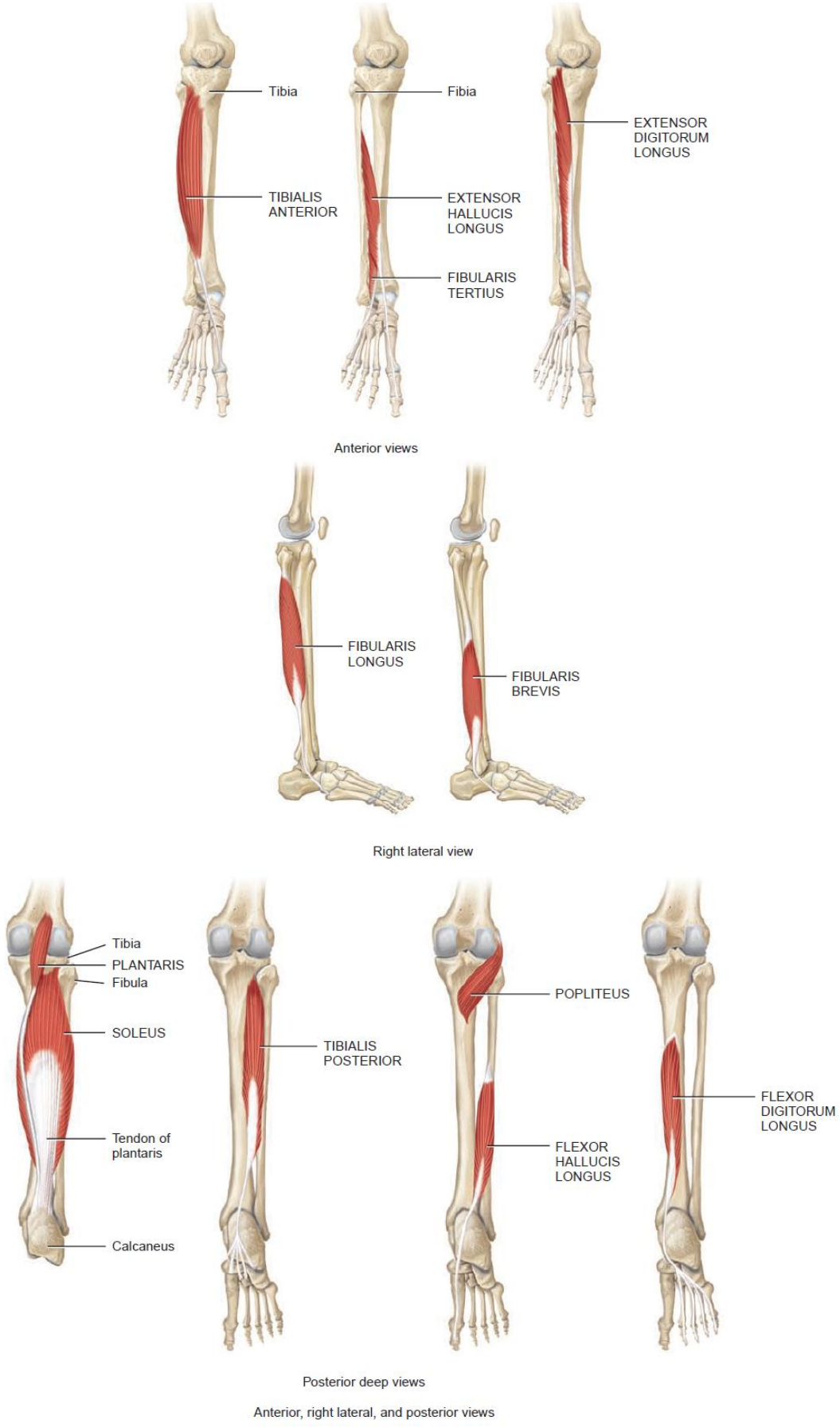


Figure 2.5 Muscles of the leg that move the foot and toes (Tortora and Derrickson, 2008)

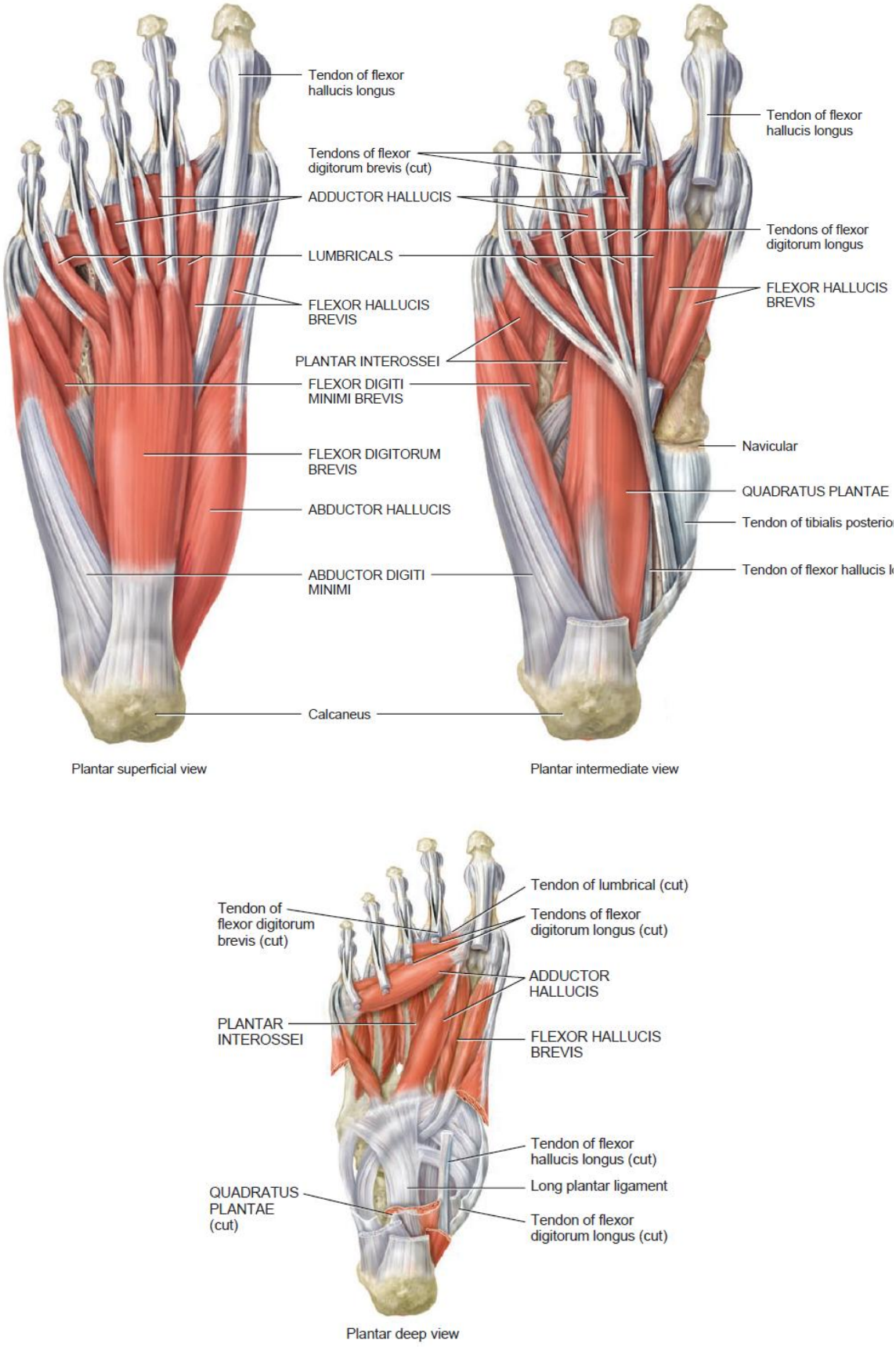


Figure 2.6 Intrinsic muscles of the leg that move the toes (Tortora and Derrickson, 2008)

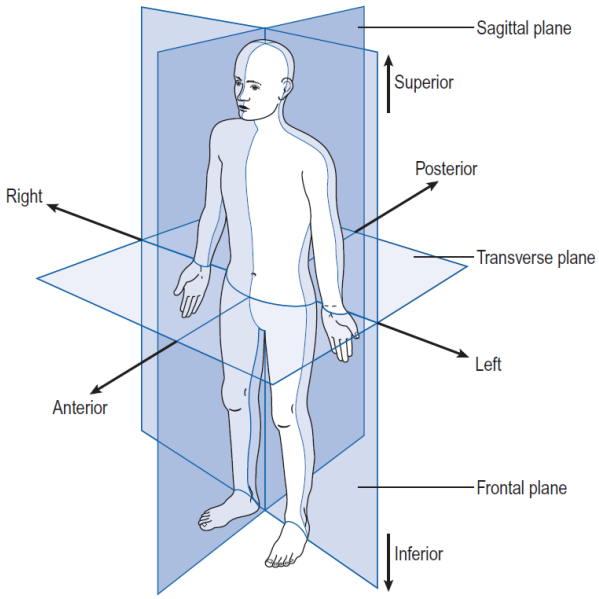
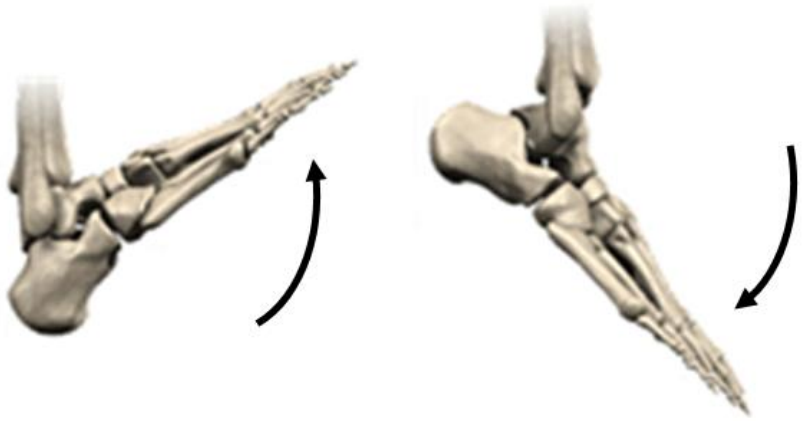


Figure 2.7 Anatomical position, with three reference planes and six fundamental directions (Tortora and Derrickson, 2008)

The planes referred in Fig. 2.7 are useful to define the joints movements. Most joints only enable movement in one or two of the planes. The possible movements are listed below and illustrated in Figs. 2.8, 2.9 and 2.10, for the case of the ankle-foot complex.

1. Flexion and extension occur in the sagittal plane. At the ankle joint, these movements are called dorsiflexion and plantarflexion, respectively (see Fig. 2.8);
2. Abduction and adduction take place in the frontal plane;
3. Internal and external rotation, also called medial and lateral rotation, take place in the transverse plane (Tortora and Derrickson, 2008).



(a) **(b)**

Figure 2.8 Ankle movements. **(a)** Dorsiflexion and **(b)** plantarflexion

The ankle joint is a major weight bearing hinge joint connecting the foot to the leg and mostly responsible for dorsiflexion (toes up, heel down) (see Fig. 2.8a) and plantarflexion (toes down, heel up) (see Fig. 2.8b). However, it also allows a slight movement in the transverse plane during plantarflexion, resulting in instability while in this position. The ankle joint is essential to the normal locomotion and the minimum range of motion necessary for a normal gait cycle is from 10º of dorsiflexion to 20º of plantarflexion (Flores et al., 2006, Tortora and Derrickson, 2008, Moreira, 2009).

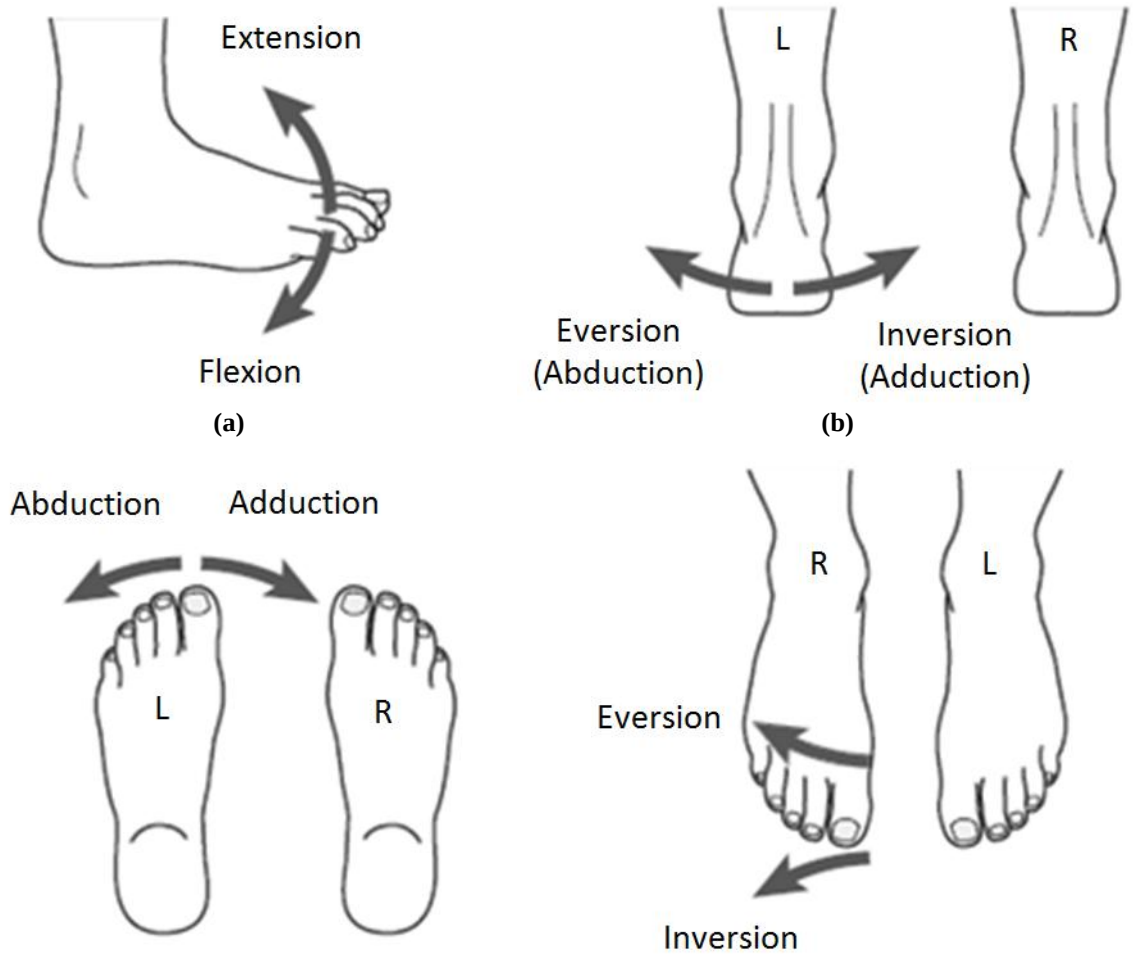


Figure 2.9 Movements of the foot. **(a)** Toe motion, **(b)** hindfoot motion, **(c)** and **(d)** forefoot motion plantarflexion (Tortora and Derrickson, 2008)

The ankle-foot complex can also perform combined movements, in order to provide both flexibility and stability during gait, such as supination and pronation (see Fig. 2.10) which are allowed by the subtalar joint. As supination (see Fig. 2.10a) can be seen as the equivalent to a simultaneous inversion, plantarflexion and adduction, pronation (see Fig. 2.10b) is comparable to simultaneous eversion, dorsiflextion and abduction (Abboud, 2002, Tortora and Derrickson, 2008).

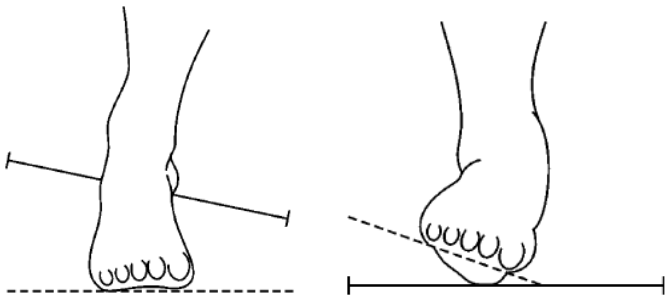


Figure 2.10 Movements of the ankle-foot complex. **(a)** Supination and **(b)** pronation (Abboud, 2002)

2.3 Human gait description

Normal human walking can be defined as a process of locomotion involving the use of the two legs, alternately, to provide support and propulsion, with at least one foot in contact with the ground at all instants (Hunt and Crossley, 1975, Whittle, 2007). The terms walking and gait are often used interchangeably but they differs slightly since the term walking refers to the locomotion process itself and the term gait refers to the "the manner or style of walking". Thus, it makes more sense to compare gait patterns than walking (Whittle, 2007).

The normal gait is a concept hard to describe since it comprises both feminine and masculine genders and a wide range of extremities of body geometry. Therefore, the normal gait is the one that is within the normal limits considered appropriate for each sex and age (Rose and Gamble, 2005, Whittle, 2007).

The human walking is a periodical movement and the gait cycle is defined as the time interval which separates two successive occurrences of one of the repetitive events of walking. It is conventionally accepted that the gait cycle starts, and ends, with the initial contact of the right foot. During the gait cycle, seven major events can be identified, namely:

1. Initial contact or heel strike (HS)
2. Opposite toe off
3. Heel rise
4. Opposite initial contact
5. Terminal contact or toe off (TO)
6. Feet adjacent
7. Tibia vertical

(1. Next cycle initial contact)

These seven events are illustrated in the sequence of Fig. 2.11. Thus, the gait cycle is divided in seven periods, which are described in Tab. 2.1 and are grouped in two distinct phases: stance and swing phases (see Fig. 2.11). The stance phase, also called support phase or contact phase lasts from heel strike to toe off, while the swing phase lasts from toe off to heel strike.

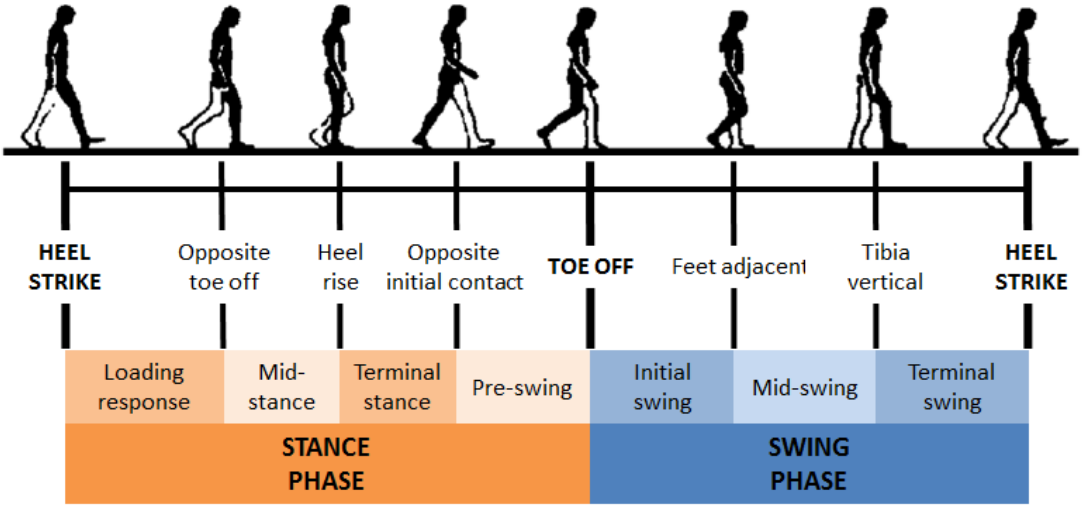


Figure 2.11 Typical normal gait cycle by the right leg (black) illustrating the major events and phases of gait, adapted from Winter (2009)

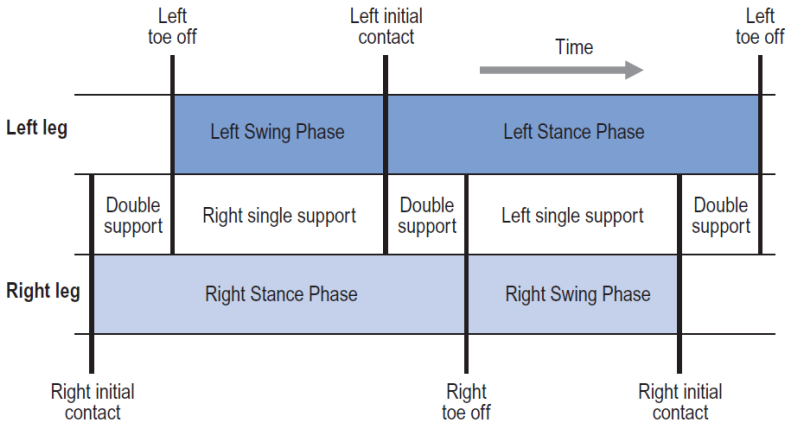


Figure 2.12 Timing of single and double support phases during both legs gait cycles (Whittle, 2007)

In each gait cycle, there are also two period of double support, i.e., periods of time when both feet are in contact with the ground, as it can be observed in Fig. 2.12. The stance phase usually lasts about 60% of the cycle, the swing phase about 40% and each period of double support about 10% (Whittle, 2007).

It must be noticed that the terms heel strike and toe off can only be applied to the non-pathological gait. Initial and terminal contact are often used in the bibliography,

since in pathological gait the heel and the toes may not be the first and last part of the foot to touch and leave the ground, respectively.

Table 2.1 Sub phases of the gait cycle with particular emphasis on the ankle joint (Whittle, 2007)

Stance phase	1. Loading response	Loading response is the double support period between heel strike and opposite toe off. During this period, the foot is lowered to the ground by plantarflexion and the GRF increases rapidly in magnitude, as total plantar contact is reached. Usually, it lasts the first 10-12% of the cycle.
	2. Mid-stance	Mid-stance is the period between opposite toe off and heel rise. During this phase, the shank is rotating forward about the ankle joint, as the foot sole remains in full contact with the ground. Thus, the ankle angle changes from plantarflexion to dorsiflexion. It represents about 18% of the gait cycle.
	3. Terminal stance	Terminal stance starts at heel rise (or heel off) and ends when the opposite heel strike occurs. At the end of terminal stance, the dorsiflexion reaches its maximum value. It represents 20% of the gait cycle.
	4. Pre-swing.	Pre-swing starts at the opposite heel strike and ends when the stance limb toe off occurs. It is the second double support period of the gait cycle. During this period, the ankle movement changes into plantarflexion and the peak of ankle plantarflexion occurs at toe off. Pre-swing lasts about 10% of the gait cycle.
Swing phase	5. Initial swing	Initial swing begins at toe off and continues until the maximum knee flexion occurs, which coincide with feet adjacent position, i.e., when the swing foot overtake the stance foot. During the swing phase, the ankle is moving from a plantarflexed position at toe off towards a neutral or dorsiflexed attitude in terminal swing. It lasts about 15% of the gait cycle.
	6. Mid-swing	Mid-swing is the period between feet adjacent and tibia vertical. At this phase, the ankle attitude becomes less important and it may be anywhere between a few degrees of plantarflexion and a few degrees of dorsiflexion. Mid-swing represents 10% of the gait cycle.
	7. Terminal swing.	Terminal swing is the final phase of the swing phase. It is the deceleration phase where the knee fully extends in preparation for heel strike. In this phase, the ankle muscles activity increases to hold the ankle in position in anticipation of the greater contraction forces developed during the loading response. Terminal swing duration is about 15% of the gait cycle.

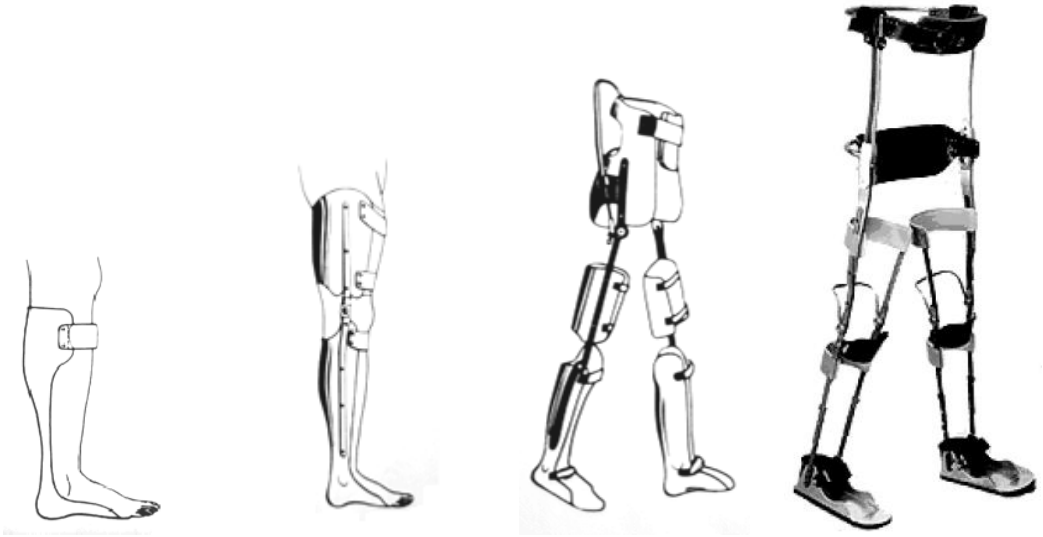
2.4 Lower limb orthoses

Orthoses are medical devices that support or correct the function of a limb or the torso (Edelstein and Bruckner, 2002). They are also known as braces or caliper. In particular, a lower limb orthosis is an external device that is applied or attached to a lower limb. Its function is to control motion, provide support, reduce pain, correct flexible deformities or prevent the progression of fixed deformities (Alexander et al., 2011).

Orthoses are usually named according to their location in the body. There are four major types of lower limb orthosis (Edelstein and Bruckner, 2002), namely:

- Ankle-foot orthosis – AFO (see Fig. 2.13a);
- Knee-ankle-foot orthosis – KAFO (see Fig. 2.13b);
- Hip-knee-ankle-foot orthosis – HKAFO (see Fig. 2.13c);
- Trunk-hip-knee-ankle-foot orthosis – THKAFO (see Fig. 2.13d) (Edelstein and Bruckner, 2002)

In the context of the present work, AFOs are considered only.



(a) **(b)** **(c)** **(d)**

Figure 2.13 Lower limb orthoses. **(a)** AFO (Advanced Orthotic Designs Inc., 2011), **(b)** KAFO (Advanced Orthotic Designs Inc., 2011), **(c)** HKAFO (Advanced Orthotic Designs Inc., 2011) and **(d)** THKAFO (ProWalk GmbH, 2009)

2.5 Ankle-foot orthosis (AFO)

An AFO is usually an orthosis that covers portion of the foot and the leg (Harris et al., 2008). It features a flat shoe sole for the foot, spans the ankle joint and covers the

lower leg (Edelstein and Bruckner, 2002, Vasconselos, 2010) (see Fig. 2.14). An essential element of AFOs is an anterior strap located just below the knee which secures the leg against the posterior calf shell (Edelstein and Bruckner, 2002).

Figure 2.14 Ankle foot orthosis Knit-Rite SmartKnit® AFO Liner (Knit-Rite, 2010)

AFOs are passive medical devices, i.e., they do not have any kind of energy supply. The term AFO should not be confused with AAFO that refers to an active ankle-foot orthosis. The AAFO consists of a generic AFO endowed of an electromechanical device that controls the ankle movements (Silva et al., 2010), thus it can force the patient ankle to assume the normal kinematic of the human gait.

AFOs can be found in pre-manufactured versions but for specific pathologies a proper custom-made solution is required. This is why the orthotic devices have evolved based on accumulate experience. Nowadays, researchers and engineers are trying to apply medical knowledge, material science and biomechanical principles to create orthoses that respond to the specific needs of every patient.

For long, AFO properties, such as rigidity and buckling, had caused problems (Golay et al., 1989). More recently, some studies using a 3D FEM of the AFO and the foot were performed in order to study the AFO behavior and to determine the maximum tensile stress and its location during gait cycle (Chu et al., 1995). Other authors developed an alternative method to evaluate the properties of the orthoses, such as the fatigue strength (Lai et al., 2010) or the stiffness and neutral angle around the ankle and metatarsal-phalangeal joints (Bregman et al., 2009).

In a clinical perspective, much work has been done on the evaluation of the effect of the orthosis support on gait analysis of patients. The patterns of motion of the pelvis, hip and knee of children affected by different degrees of lumbosacral

myelomeningocele with and without orthoses had been illustrated and compared (Vankoski et al., 1995). The frequency of adjustments and replacement of the orthosis has also been evaluated (Supan and Hovorka, 1995). There are some works on how the use of an AFO changes the kinematic and the kinetic of human gait (Huang et al., 2006, Silva et al., 2011). The influence of the AFO in the patterns of muscular activity has been investigated in order to understand the neuromuscular adaptation associated with the use of the orthotic device (Crabtree and Higginson, 2009). The contact forces (Silva et al., 2010) and the pain pressure threshold (Coutinho et al., 2011) in the orthosis/lower limb interface have also started to be explored to ensure the patient comfort.

AFOs can be classified according to the material they are made of, as Fig. 2.15 shows.

(a) **(b)** **(c)** **(d)**

Figure 2.15 Ankle foot orthoses. **(a)** Metal AFO (OrthoMedics, 2011), **(b)** plastic AFO (OrthoMedics, 2011), **(c)** hinged plastic AFO (OrthoMedics, 2011) and **(d)** carbon fiber AFO (Kinetic Research Inc., 2011)

Metal AFOs consist of a shoe or foot attachment, an ankle joint, two metal uprights (medial and lateral) with a calf band connected proximally (see Fig. 2.15a). They are indicated for several specific pathologies, including the insensate foot, the foot with fluctuating edema, or when the need for adjustability or progressive changes in the device are indicated (Cooper, 2006, Alexander et al., 2011).

There are two types of ankle joints used in metal AFOs:

- Single-channel ankle joints, which provide dorsiflextion assistant and a plantarflexion stop;
- Dual-channel ankle joints, which assist the foot both in the dorsiflexion and plantarflexion directions (Cooper, 2006).

Plastic AFOs (see Fig. 2.15b) are the most common type of AFO. They are mostly made from a thermoplastic material, polypropylene (PP), for the structural

components and Velcro straps for tightening (Silva et al., 2010). They can be fabricated from a cast or molding of the patient's limb. The first approach may be suitable for short-term use but the second-one is better for durable use (Cooper, 2006), allowing choosing the plastic type, color and thickness. The general features of a plastic AFOs would include the trimlines (degree of rigidity), degrees of dorsiflexion, and foot plate design (Cooper, 2006).

The main characteristic of plastic AFOs is the posterior leaf spring (PLS) design. This the most common form of AFO with a narrow calf shell and a narrow ankle trim line behind the malleoli. It is typically set in 5-7º of dorsiflexion with very low-profile three-quarters length footplate. The PLS is used for compensating flaccid footdrop by resisting ankle plantarflexion at heel strike and during swing phase (Edelstein and Bruckner, 2002, Cooper, 2006, OrthoMedics, 2011).

Plastic AFOs can also incorporate a hinged joint at the ankle (see Fig. 2.15c) which will allow some dorsiflexion and a limited plantarflexion. However, hinged-AFOs are less adjustable than metal AFO joints. The footplate design can incorporate three-quarter length, which stops just before the metatarsal heads for easier access into shoes, or a full length footplate with padding, which is generally used for the most spastic or most vulnerable foot (Cooper, 2006).

There are also many variants of plastic AFOs, designed for specific diseases (see Fig. 2.16) (Edelstein and Bruckner, 2002, Alexander et al., 2011). However, in this work, it is not relevant to describe them because of their specificity.

Carbon AFOs (see Fig. 2.15d) have been widely used during the last decade. Carbon fiber is a material extremely lightweight, and durable, however it is not adjustable and does not fit perfectly in the limb. This style of AFO is best used for isolated foot drop (OrthoMedics, 2011).

2.6 Requirement for AFOs

AFOs are prescribed and used to restore normal motion or to constrain and inhibit abnormal motion (Chu, 2001). They can be used to improve the base of support of patients with balance perturbation but they also improve ankle kinematics during stance phase, increase step and stride length, decrease cadence, and decrease energy costs in walking, while improving walking, running and jumping skills (Chen et al., 1999, Brehm et al., 2008, Harris et al., 2008).

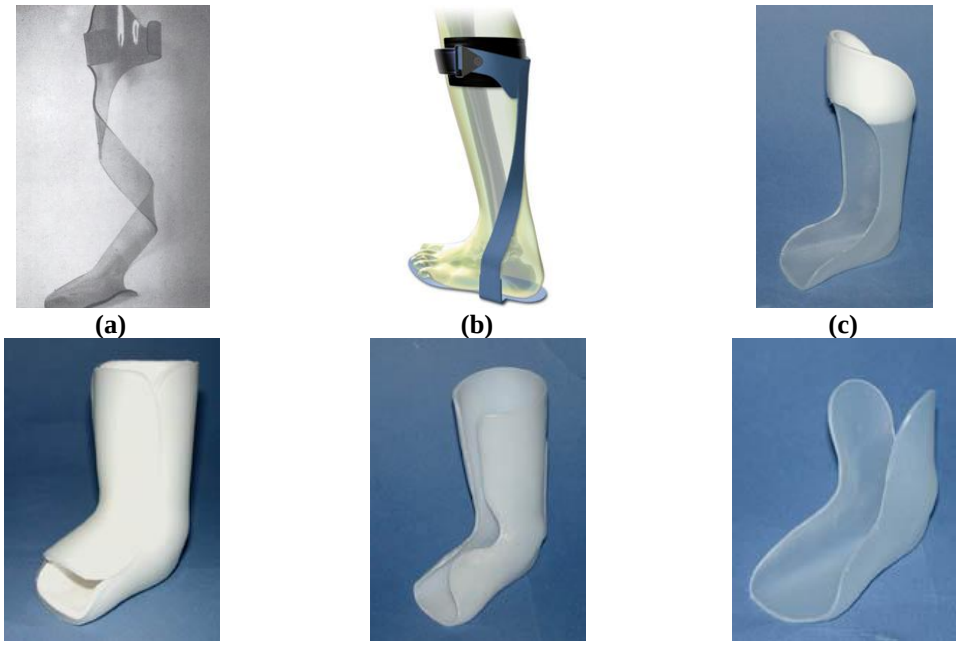


Figure 2.16 Variants of plastic AFO orthoses. **(a)** Spiral AFO (Edelstein and Bruckner, 2002), **(b)** hemispiral AFO (Lower Extremity Review Magazine, 2011), **(c)** ground reaction AFO (NovitaTech, 2010), **(d)** bi-valved AFO (NovitaTech, 2010), **(e)** circumferential AFO (NovitaTech, 2010) and **(f)** supramalleolar AFO (NovitaTech, 2010)

AFOs are indicated for a great variety of pathologies which may be neurological, vascular or orthopedic (see Tab. 2.2). Many patients suffering from these pathologies have lack of control of the lower limbs, necessity of lowering the pressure on the feet or need for support (Silva et al., 2010). A very common symptom in these cases is the dropfoot.

Dropfoot is an abnormal neuromuscular disorder characterized by a steppage gait that affects the patient's ability to raise their foot at the ankle, and is further characterized by an excessive and uncontrolled plantarflexion, an inability to point the toes towards the body (dorsiflexion) or move the foot at the ankle inwards or outwards. The dropfoot motion will lead to toe dragging during the swing phase of the gait cycle and result in pain and weakness. Moreover, numbness may accompany loss of function (Chu, 2001, Jamshidi et al., 2009).

AFOs can also be used to provide support in cases of general weakness and positional support for patients with Excessive Muscle Tone, Paralysis, or Congenital Deformity. Finally, they can be applied to immobilize the foot/ankle in cases of Charcot Feet, Fracture, Arthritis, or Wound Management (OrthoMedics, 2011). However, the most common application of AFOs is to provide support for dropfoot or for ankle instability associated with several conditions some of which include stroke, spina bifida,

cerebral palsy, Lou Gehrig's Disease, multiple sclerosis, paraplegia or poliomyelitis (Advanced Orthotic Designs Inc., 2011).

Table 2.2 Pathologies associated with the use of AFOs (Chu, 2001, Silva et al., 2010, Advanced Orthotic Designs Inc., 2011, OrthoMedics, 2011)

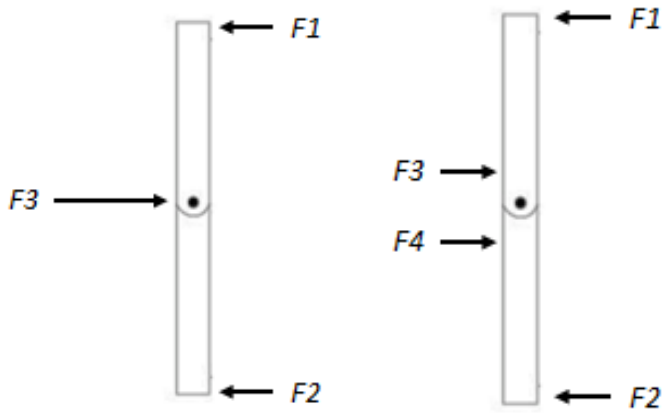
List of Pathologies
▪ Cerebral palsy
▪ Stroke
▪ Lou Gehrig's disease
▪ Multiple sclerosis
▪ Poliomyelitis
▪ Paraplegia
▪ Paralysis (Hemiplegia)
▪ Nerve Damage
▪ Spinal cord injuries
▪ Excessive muscle tone
▪ Traumatic brain injuries
▪ Neuropathy
▪ Charcot Feet
▪ Congenital deformities
▪ Injuries/Fractures
▪ Joint diseases (e.g. Arthritis)
▪ Muscular dystrophy
▪ Spina bifida
▪ General weakness
▪ Wound management

2.7 Biomechanics of AFOs

In order to maintain the anatomical joints in their proper positions and to restrain abnormal movement, orthosis design is normally based on two types of forces system: the three point pressure (3PP control) and the GRF control (Edelstein and Bruckner, 2002, Pakistan Academy of Orthotists & Prosthetists, 2009).

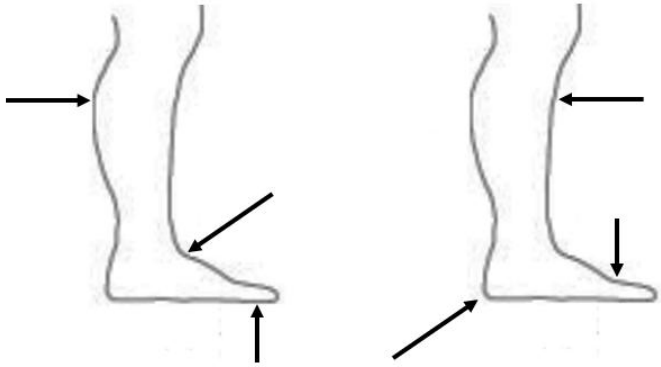
In the first case (3PP), it is intended to block or restrain the rotation of two body segments about the anatomical joint that unites them. An example is given in Fig. 2.17a on how the rotation of an articulation can be prevented by applying three forces: one at the free end of each segment (*F1* and *F2*) and a third force directly at the revolute joint (*F3*). A variation of the 3PP control system, often used in orthotic practice, is the four point pressure system (see Fig. 2.17b). In this system, the force *F3* is replaced by two forces (*F4* and *F5*) to decrease the pressure applied directly at the anatomical joint.

In the case of AFOs, the 3PP control system is applied to prevent motion the anatomical joints of the ankle foot complex. Figure 2.18 shows the forces developed by an AFO on the ankle joint to restrain plantarflexion (see Fig. 2.18a) and dorsiflexion (see Fig. 2.18b).



(a) **(b)**

Figure 2.17 Graphical representation of the **(a)** three and **(b)** four point pressure systems (Pakistan Academy of Orthotists & Prosthetists, 2009)



(a) **(b)**

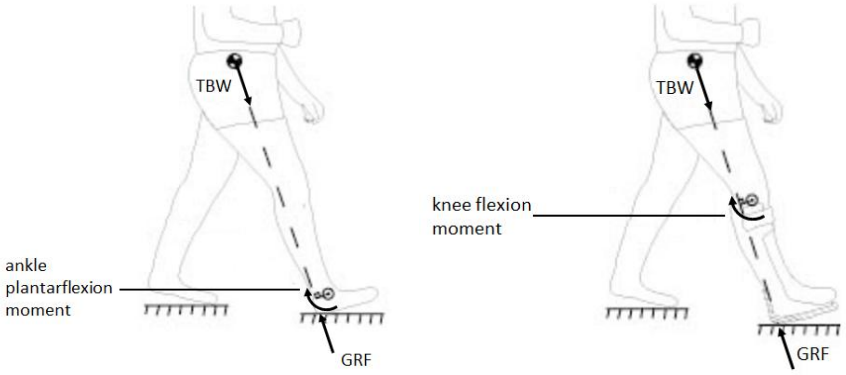
Figure 2.18 Three point pressure systems of an AFO developed to prevent **(a)** plantarflexion and **(b)** dorsiflexion (Pakistan Academy of Orthotists & Prosthetists, 2009)

The GRF control aims to rectify the motion of a body segment and/or a joint using (or not) an orthopedic device during the stance phase, i.e., when the foot contacts the floor.

At heel strike, the heel hits the floor and a GRF is generated as an equal and opposite force. If the total body weight is not aligned with the ankle joint, the GRF will create a plantarflexion moment at the ankle joint (see Fig. 2.19a). However, as many patients have a foot disability, dropfoot, they cannot dorsiflex their foot back to a neutral position. For this reason, the AFO is used to restrain the plantarflexion at the initial contact in the gait cycle, and the GRF is transferred to the next free joint in the kinematical chain – the knee joint (see Fig. 2.19b), creating a flexion moment and preventing knee hyperextension (Pakistan Academy of Orthotists & Prosthetists, 2009).

In AFOs design, changing the lever arm length or the surface area can increase the patient's comfort. Since the moment *M* developed at the ankle joint can be calculated

through the equation $M = F.l$, increasing the length l will reduce the force F developed between the orthosis and the patient limb. The pressure exerted on the patient skin is also a matter of concern and may reach relatively high values. However it can be easily decreased by an increase of the surface area as the pressure is inversely proportional to the surface area, $P = F/A$ (Pakistan Academy of Orthotists & Prosthetists, 2009).

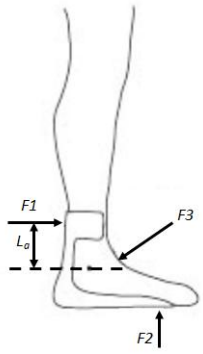
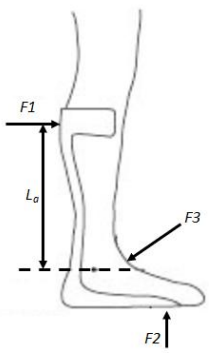
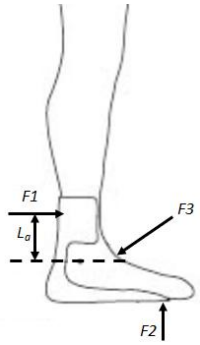
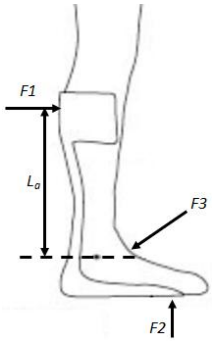


(a) **(b)**

Figure 2.19 GRF control in the sagittal plane at heel strike **(a)** without AFO and **(b)** with AFO (Pakistan Academy of Orthotists & Prosthetists, 2009)

Table 2.3 shows the different combinations of lever arm length and calf shell surface area. The optimum solution to assure both efficiency and patient comfort is to maximize the length of the lever arm and the calf surface area.

Table 2.3 Effects of lever arm and surface area in AFOs, adapted from Pakistan Academy of Orthotists & Prosthetists (2009)

		Lever arm length	
		short	*increased*
Surface area	*Small*		
	increased		

CHAPTER 3

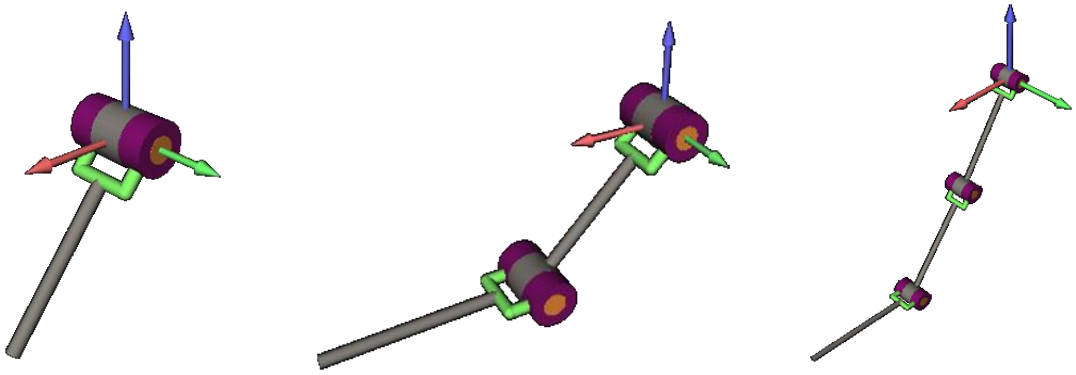
MODELLING THE HUMAN BODY

The following chapter begins by presenting the software used, MOBILE, and the first simple models developed to get familiar with this programming environment. The MBS model is described, with particular emphasis on the contact foot model. Relevant contact foot models existing in the literature are referred and described in order to contextualize the developed model.

3.1 MOBILE description

MOBILE code is an object-oriented programming package designed for the modeling of MBS (Kecskeméthy, 1999). It is a software developed by Professor Andrés Kecskeméthy and is still being developed and improved at the Duisburg-Essen University. It allows for an intuitive 3D representation of mechanical elements that can transmit motion and forces, being the modeling of mechanical systems as executable programs. The implementation is portable and efficient since it is based on the object-oriented programming language C++, and the building-block system design is open, allowing the user to extend the provided library in any direction. To facilitate the visualization of the simulation, it is also possible to import geometries from *.so* (UNIX®, The Open Group) or .iv (Open Inventor™, Silicon Graphics Incorporated) files (Kecskeméthy, 1999).

The first work completed on MOBILE was the modeling and simulation of simple models, namely pendulums (see Fig. 3.1). The first model created was the simple pendulum described in the User's Guide (see Fig. 3.1a). This model revealed to be useful to consolidate the concepts learnt during the lecture of the MOBILE User's Guide (Kecskeméthy, 1999). With the modeling of these concepts, the basic MOBILE objects and the structure of the program becomes clear and the user become more familiar and confident with the programming in C++.

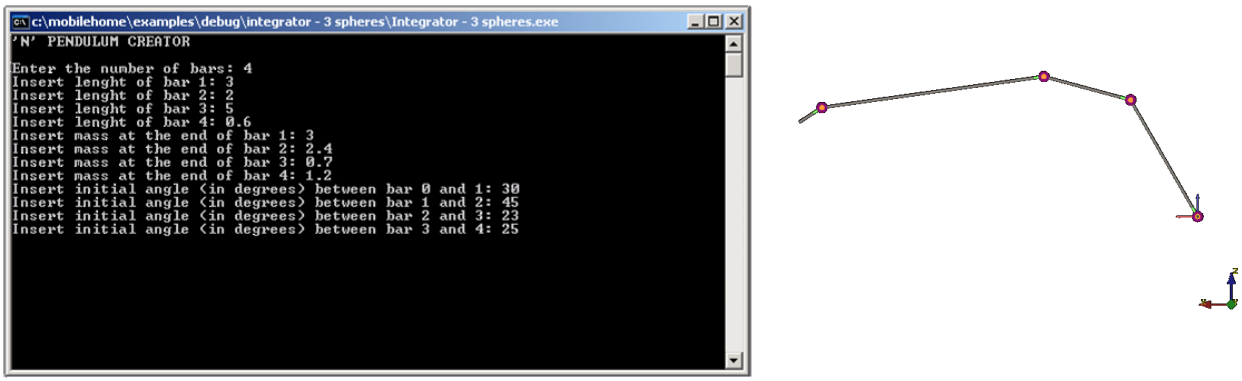


(a) **(b)** **(c)**

Figure 3.1 (a) Simple pendulum, **(b)** Double pendulum and **(c)** Triple pendulum

After the simple pendulum analysis (see Fig. 3.1a), a double (see Fig. 3.1b) and a triple pendulum (see Fig. 3.1c) were modeled, only with the addition of rigid bodies, joints and masses to the kinematical chain.

Finally, an N'pendulum was created. In this model, all the parameters of the model are inserted by the user (number of bars, lengths, masses and initial conditions), as it is represented in Fig. 3.2. The code for the N'pendulum is attached on Appendix I.



(a) **(b)**

Figure 3.2 N'pendulum creator. **(a)** Example of parameters inserted by the user and **(b)** Initial conditions for the parameters set in (a)

3.2 Development of the full-body model

A biomechanical model of the human body was constructed in MOBILE. This model is bi-dimensional and was constructed in the sagittal plane, since the major amplitude of movement and forces occur on this plane during the gait cycle (Silva et al., 2010).

The MBS model illustrated in Fig. 3.3 consists of a pair of legs that represent the human lower limbs, totalizing 9 rigid bodies, 10 revolute joints and 2 prismatic joints. Each leg was built up by 4 rigid bodies (thigh, shank, foot and toes) and an additional body was created to represent the upper-body. This last body is called HAT and has the total mass of the upper-body (Winter, 2009). All the bodies are linked by revolute joints. Three additional DOFs were added at the hip (translations in two directions and rotation in the sagittal plane, in this order), so the model can move freely in the plane.

The first simulation procedure with this model consisted of driving all the joints with time functions. In this way, the whole system is rheonomic since all the DOFs were guided. The time functions were obtained in a gait lab. This model, called "*Reader*", is nothing more than a player that allows the user to visualize the measurements in MOBILE. The code for the "*reader.cpp*" is included in Appendix II.

The second model is called "*Integrator*" since it contains an integrator prepared for forward dynamics. This model contains a contact model between the foot and the floor and the 3 DOFs at the hip are now computed as well at the right ankle joint and the metatarsal joints. A full description of this model can be found in Section 4.4.

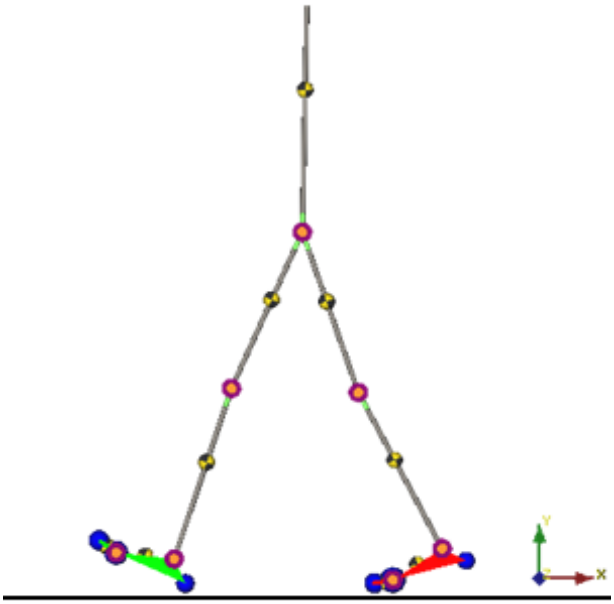


Figure 3.3 MBS model created in MOBILE for gait analysis

3.2.1 Parameters of the model

The parameters of the model, such as the lengths of the rigid bodies, their masses, the location of the center of masses (CM) and the moments of inertia had to be calculated and inserted as input data.

In a first approach, the data used by (Silva et al., 2010) were applied directly to the model, but the necessity for more accurate results required the geometric data from the subject whose gait was being analyzed.

The length of each segment was calculated as the mean value of its length during the gait cycle because the values obtained in each frame change due to skin motion that changes the distance between the markers. The masses, the position of the CMs and the moments of inertia of each segments were calculated with the Winter's coefficients (Winter, 2009). These coefficients state that each segment has a constant mass that can be calculated as a percentage of the total mass of the body. The position of each CM is at a known percentage of the body length away from the proximal end of the extremity, as Tab. 3.1 illustrates.

The moments of inertia can also be calculated with the mass of each segment and the radius of gyration of each segment (see Tab. 3.1), with Equation (3.1).

$$I_0 = m\,\rho_0^2 \tag{3.1}$$

where

I_0 is the moment of inertia about the CM;

m is the mass of the segment;

ρ_0 is the radius of gyration.

A script (*anthropometric.m*) was created in MATLAB in order to compute all these parameters automatically and save them in *.dat* files to be read by the MOBILE program. This script is attached in Appendix III.

Table 3.1 Anthropometric parameters of the MBS model (Winter, 2009)

Segment	Definition	Segment Weight/Total Body Weight	Proximal CM Position/Segment Length	Radius of Gyration/Segment Length CM
HAT	Greater trochanter /glenohumeral joint	0.678	0.626	0.496
Thigh	Greater trochanter/femoral condyles	0.100	0.433	0.323
Shank	Femoral condyles/medial malleolus	0.100	0.433	0.302
Foot	Lateral malleolus/head metatarsal	0.013	0.500	0.475
Toes	Head metatarsal/distal phalanx I extremity	0.0015	0.500	0.000

3.2.2 Foot geometry

The foot geometry considered in this work was created based on the data obtained in the gait lab. It is a rigid body defined by 3 points: ankle joint, heel and metatarsal joint. The positions of the frames defining these points were assumed to be the positions of the markers used in the data acquisition in the gait lab. Figure 3.4 shows the geometry, a triangular prism, used for the visualization of the foot in MOBILE. This prism was simply obtained by uniting the foot markers (Heel, Ankle and Toe) and extruding the resulting triangle. The *.so* code wrote for the foot geometry definition is attached in Appendix IV.

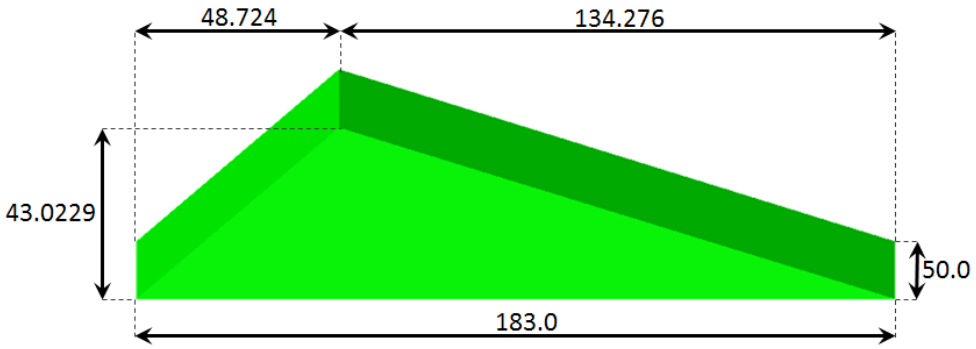


Figure 3.4 Foot geometry used in the simulation. Dimensions are in mm.

3.3 Foot model

3.3.1. Literature review on contact foot models

As described in Section 3.2.2, a foot geometry had to be created and inserted as part of the MBS model. Since the beginning of forward dynamics human gait simulations, foot models have been a major concern for biomechanical engineers. Usually, the contact force computation is computationally heavy and a simple, yet accurate foot contact model is very desirable (Millard et al., 2008). Several foot models using different geometries and contact and friction properties have been employed. However, only the most representative models are discussed below.

Millard et al. (2008) proposed three simple contact models based on sphere-plane contacts, which are presented in Fig. 3.5. The first model consists of only one rigid body with two spheres at the heel and the metatarsal joint. The second one is composed of two rigid bodies and has an additional DOF. The supplementary body was added to improve the normal ground force profile and has a sphere at the toe tip. In order to improve the accuracy of the results, a third model was suggested incorporating one more sphere to improve midfoot contact. During simulations, normal forces were computed using the Hunt-Crossley point contact model (Hunt and Crossley, 1975). Friction forces were initially computed using a Coulomb friction model which was replaced by a bristle friction model.

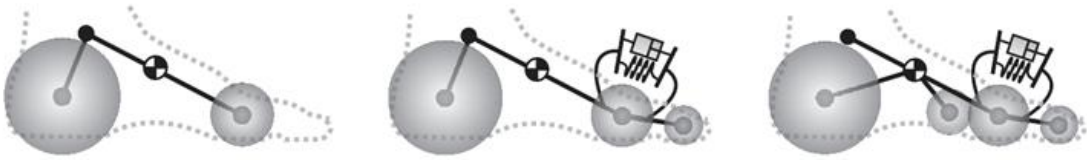


Figure 3.5 Milliard foot contact models (Millard et al., 2008)

Moreira et al. (2009) presented a three-dimensional foot contact model made of two rigid segments, connected by a revolute joint with a torsional spring-damper system (see Fig. 3.6). The model included a total of 9 spheres (6 at the plantar surface and 3 at the toes). The normal forces were computed using the Hunt-Crossley model and the friction model included the standard Coulomb friction and a viscous friction component. This model showed promising dynamics results and proved to be appropriate for simulation purposes.

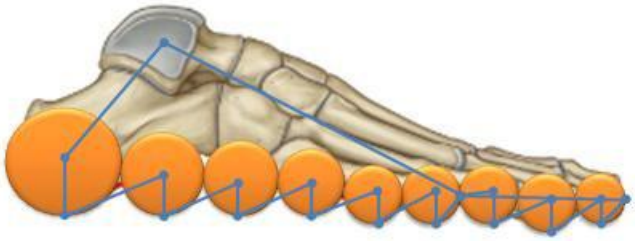


Figure 3.6 Moreira's foot contact model (Moreira et al., 2009)

As most of the foot-contact models are based on sphere-plane approaches, lately, in 2011, Kecskeméthy presented an alternative (Kecskeméthy, 2011). This model is made of two bodies connected by a spring-damper element and used two cylinder-plane contact elements for the forefoot and the heel contact (see Fig. 3.7). Normal forces are computed using the Hunt-Crossley model and the tangential forces (sticking and sliding) were computed using the Coulomb's law of friction.

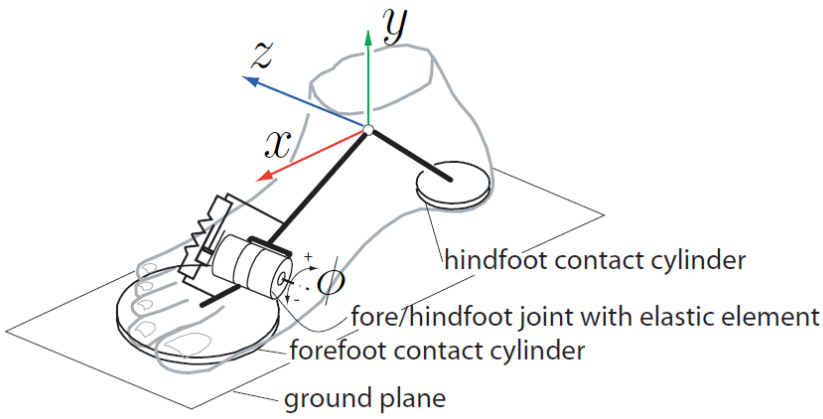


Figure 3.7 Two cylinder-plane foot contact model (Kecskeméthy, 2011)

3.3.2. Development of the foot model

In order to use forward dynamics, it was necessary to develop a contact model between the foot and the floor. A set of three spheres-plane interaction was developed

since it is the simplest model and it allowed reaching satisfying results with efficient computation.

The contact model used between the feet and the floor is the Hunt-Crossley's contact model with a sphere-plane geometry (see Fig. 3.8). A spring-damper was added at the metatarsal joints in order to provide more adaptability to the feet. In this way, the feet have a greater freedom of movement than when they were guided by time-functions and simulations showed the model was more stable. Both the spring-damper and the contact model were existing objects from the MOBILE library (*MoRegImpSpherePlane* and *MoLinearSpringDamper*).

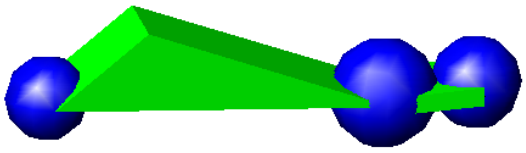


Figure 3.8 Foot geometry with identification of the spheres position used in the contact model

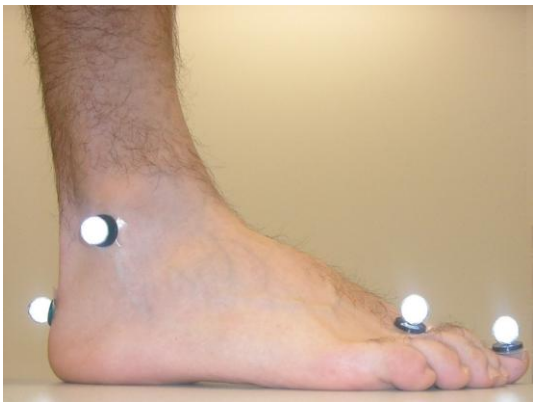


Figure 3.9 Foot with the pre-defined markers and the additional-one at the toe tip

Since the foot geometry was created based on the markers position (see Fig. 3.9), the radius of the spheres was defined as the minimum height reached by the markers during the data acquisition at the gait lab. These radiuses were calculated in MATLAB in the anthropometric script attached in Appendix III. All the remaining parameters, such as the spheres stiffness's, restitution coefficient in normal and tangential directions, sticking and sliding friction coefficient were obtained manually by an iterative method described in the Chapter 4.

3.3.3. Contact force model for sphere-plane interaction

Impact is a phenomenon that occurs when two or more bodies, which may or may not belong to a MBS, collide. The collision is characterized by an abrupt change in the MBS variables, and in particular in velocities which causes significant accelerations or decelerations (Flores et al., 2006). During impact, the surface of the contacting bodies is deformed and two distinct phases can be identified: compression and restitution. In the first one, the bodies deform in the normal direction to the impact surface until the

maximum penetration, and during restitution, the bodies start to separate until they no longer contact each other. The restitution coefficient varies between 1, for fully elastic, and 0 for a fully plastic contact (Flores et al., 2006).

The impact forces developed between the contacting bodies are a critical aspect of MBS simulation. The mechanical properties of the foot-floor interface are crucial to proceed to the forward dynamics simulation. As referred in Section 3.3.2, the contact model used in this work had a sphere-plane geometry (see Fig. 3.8), being the sphere deformable and the plane (ground) rigid. The contact class applied in the simulations was an existing MOBILE class (*MoRegImpSpherePlane.cpp*), developed by Grabner (Grabner, 2003). This class computes the normal forces using the well-known Hunt and Crossley (Hunt and Crossley, 1975), which includes both elastic and damping behavior as follows.

$$F_N = cx^n \left[1 + n.\frac{1}{|\dot{x}_i|}\left(\frac{1}{e} - 1\right)\dot{x}\right] \tag{3.2}$$

In Equation (3.2), F_N represents the contact force in the normal direction and c is the relative stiffness that depends on the geometric and material properties. The term n corresponds to the non-linear degree, and is normally to 1.5 for Hertzian contact. The coefficient e is the coefficient of restitution and the terms x, $\dot{x}$ *and* $\dot{x}_i$ represent the relative penetration depth, the velocity and the velocity at impacting time, respectively.

Equation (3.2) can be divided in two portions (Eq. 3.3), where the first one corresponds to the materials elasticity behavior, based on Hertz theory. The second part represents the dissipated energy during the impact process and depends on the impact velocity, the coefficient of restitution, as well as on x^n and $\dot{x}$ (Grabner, 2003).

$$F_N = cx^n + cx^n\dot{x}\left[n.\frac{1}{|\dot{x}_i|}\left(\frac{1}{e} - 1\right)\right] \tag{3.3}$$

For each impact, the term $n.\frac{1}{|\dot{x}_i|}\left(\frac{1}{e} - 1\right)$, also denoted as the damping constant, is calculated only at the beginning of each contact, since it remains unchanged during the impact (Grabner, 2003). Finally, it must be stated that Eq. 3.3 depends directly on the relative penetration which ensures the force value is null when there is no contact. Nevertheless, it is important to notice that not only the normal forces exist in the sphere-plane interface. Friction forces are also developed in the tangent direction and must be computed. The sticking force ($F_{T,st}$) formula was obtained by adapting the Hunt-Crossley model (Eq. 3.4).

$$F_{T,st} = -c_T|x_T|^n \left(\frac{x_T}{|x_T|} + \frac{3}{2}\frac{1-e_T}{|\dot{x}_T^-|}\dot{x}_T\right) \quad (3.4)$$

In this equation, c_T and e_T are the tangential stiffness and the tangential coefficient of restitution, respectively and x_T represents the tangential vector transpose in the tangent plane. In the 2D case, $x_T = [x\ y]^T$ denotes the deflection vector in the tangent direction. The transition from sticking to sliding is characterized a vanishing friction saturation $\triangle F_T$ (Eq. 3.5). The term μ_{st} represents the coefficient of sticking friction. When $\triangle F_T=0$, the contact force is getting out of the friction cone (see Fig. 3.10) and sliding takes place. The friction cone is a virtual geometry defining the equation $\mu_0 F_N = |F_{T,st}|$ and has the particularity that $\alpha = \tan^{-1}\mu_{st}$ (see Fig. 3.10). When the friction forces are applied inside the cone, the sphere is sticked to the plane but when they reach the surface of the cone, the sphere begins to slide.

$$\Delta F_T = \mu_{st} F_N - |F_{T,st}| \quad (3.5)$$

When sliding takes place, the dynamic friction force $F_{T,sl}$ is computed according to Coulomb's friction law (Eq. 3.6), where μ_{sl} represents the coefficient of static friction as follows.

$$F_{T,sl} = \mu_{sl} F_N \quad (3.6)$$

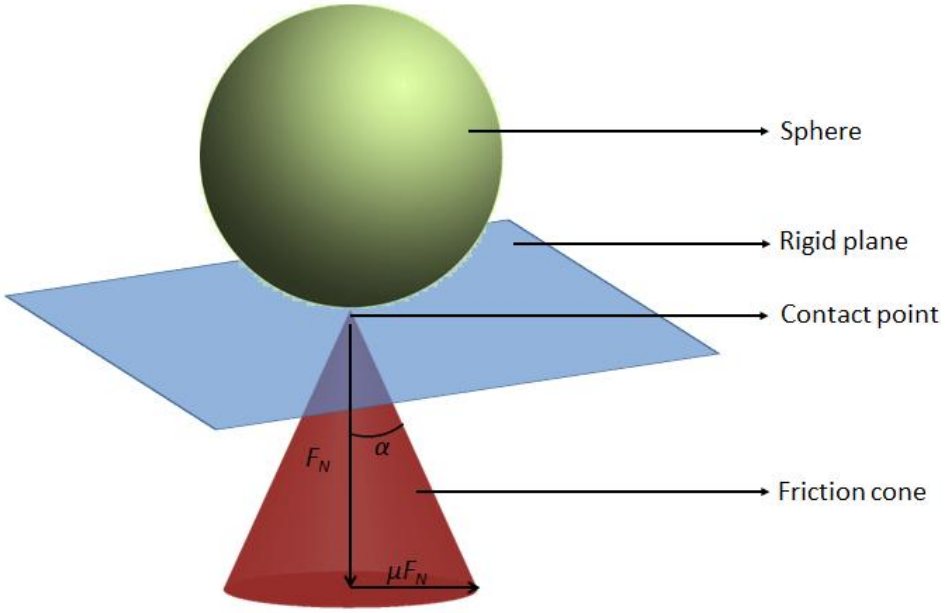


Figure 3.10 Graphical representation of the friction cone

In short, the MOBILE class used in this thesis work, computes normal forces based on the Hunt-Crossley contact model and the friction forces based on Coulomb's theory of friction (Grabner, 2003).

CHAPTER 4

METHODS, RESULTS AND DISCUSSION

This chapter deals with the methodologies employed. The data acquisition procedure in the gait lab is exposed and the use of the measured data as time functions to guide the joints is explained. The problems faced and their corresponding solutions are presented. The measured data are presented as ankle moment over ankle angle and ankle angle over time plots which are described and discussed. Simulations procedure, barefoot and with orthosis, are explained and results are exposed and discussed.

4.1 Data acquisition in the gait lab

Kinematics is the study of movement characteristics, in terms of displacements, velocities and accelerations, independently of the forces that cause this motion (Winter, 2009).

In the contest of this work, kinematic measurements were performed in a gait lab, which consist of placing a set of passive markers (reflective balls) in the subject skin, at palpable anatomical landmarks (see Fig. 4.1). The gait lab used for these measurements belongs to the Department of Mechanics and Robotics of the Duisburg-Essen University and comprises a VICON® MX 13 motion capture system with seven cameras, 2 force plates (AMTI® OR6-7-2000), and 2 camera recorders.

VICON® cameras utilize IR high-powered strobes to record the reflection of each marker. The triangulation of each marker position is possible in 3D based on the angle and time delay between the emitted and reflected signal. The software available, VICON® Nexus, labels each marker, creates a 3D-MBS model by connecting the markers and tracking their trajectories during the gait cycle analysis. The software computes the angular position, velocity and acceleration of each joint, and calculates the forces, moments and power applied at the joints by measuring the GRFs. For this purpose, the inverse dynamic approach is used.

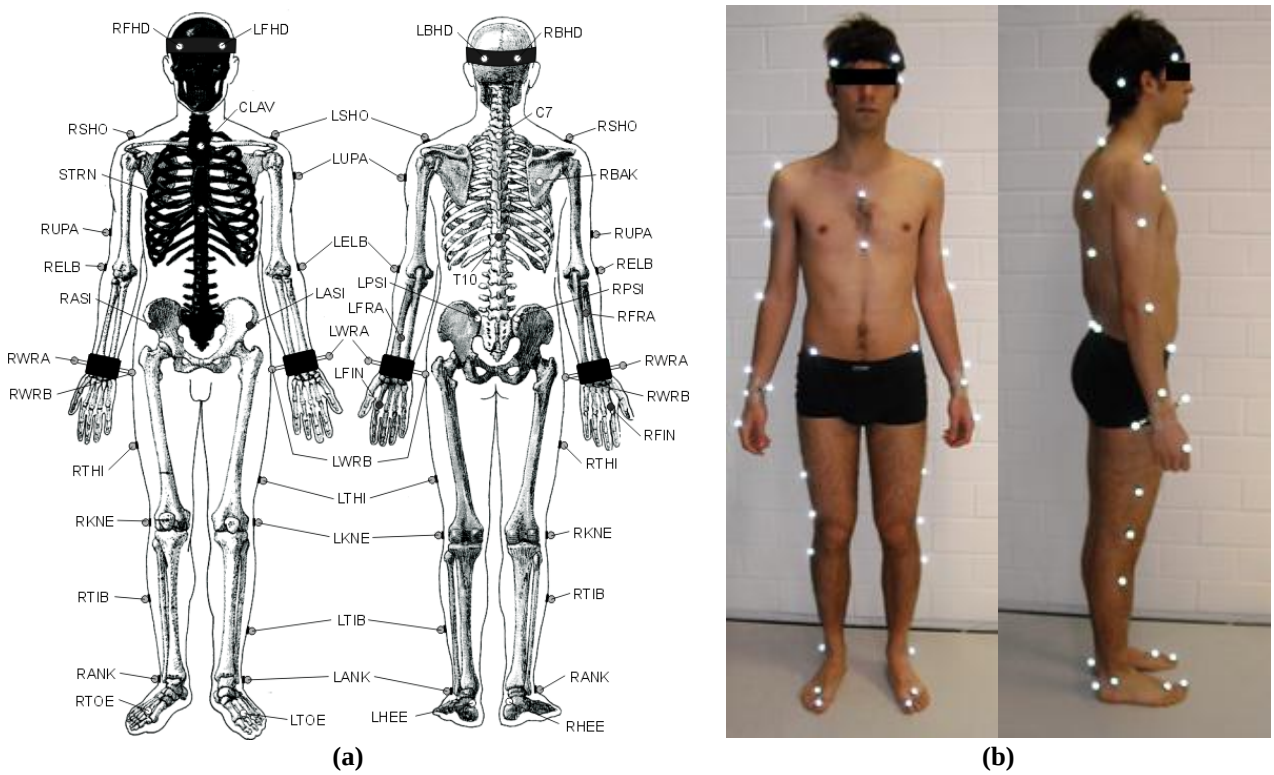
The default set acquisition properties include placing 39 markers (see Fig. 4.1a) on the subject skin (see Fig. 4.1b). However, since the human body model developed (see Fig. 3.3) had a joint to simulate the metatarsal articulation, an additional marker was added at the big toe tip. This marker was fixed in the nail of the big toe to avoid skin motion and increase the precision of the measurements, as it can be seen in Fig. 4.2.

During the gait analysis, the subject walks on a 8 meters long walkway. Measurements are recorded along with a video of the trial. The sampling frequency is

100Hz and VICON Nexus software filters automatically the trajectory of the markers, using a Woltring filtering routine, with a mean squared error (MSE) of 10. Therefore, no need of post-signal-processing is necessary. The results are exported in a matrix with the 3D coordinates of each marker in time and are exported to a *.c3d* file that can be imported in MATLAB.

Woltring filtering routine is a filter designed for spline smoothing and differentiation. It has been proved that Woltring filter is equivalent to a double Butterworth filter, with the only improvement that it can process data with unequal sampling intervals. The MSE method allows the user to define the noise level and the spline is fitted to the data points allowing the given level of tolerance (Woltring, 1986, International Society of Biomechanics, 2012, VICON, 2012).

It is important to point out that the resulting plots are not formatted in the SI units system but with the units system used by VICON Nexus (angles in degrees, lengths in m, masses in kg and forces in N).



(a) **(b)**

Figure 4.1 (a) Position of the markers and **(b)** Subject with the markers on his skin

4.1.1 Results with and without orthosis

It is known that AFOs are actuated mostly at the ankle articulation. In this way, measurements of an healthy subject were taken with and without orthoses to assess their effect on the normal human gait. The first trials were performed barefoot, while the

second measurements were made with orthoses worn in both legs. For the measurements obtained with orthoses, it was necessary to fix the orthoses to the foot and shank with adhesive tape (see Fig. 4.2). Usually, AFOs are fixed to the subject within his shoes but according to the gait lab procedures, the measurements should be done barefoot and the markers must be placed directly on the subject skin.

Figure 4.2 Fixation of the orthosis to the subject foot and shank with adhesive tape

The results obtained via inverse dynamics in the gait lab can be analyzed by plotting the ankle moment over the ankle angle, as it is plotted in Fig. 4.3. In this diagram, a clear difference between the results with and without orthosis can be observed. With the orthoses, there is almost no hysteresis in the ankle joint since the moment at the ankle increases (dorsiflexion) and decreases (plantarflexion) almost linearly with the ankle angle. Figure 4.3b shows also a similar behavior for both plantarflexion and dorsiflexion and the area inside the loop was reduced to nearly zero which shows a behavior that could be mimicked using an efficient torsional spring (Hansen et al., 2004). The measurements obtained without orthoses (see Fig. 4.3a) show that there is some damping, since the curves for the plantarflexion and dorsiflexion exhibit some hysteresis. However, with orthoses in both feet, damping can be neglected because both plantarflexion and dorsiflexion plots have very close evolutions.

Another conclusion that can be drawn from the analysis of Fig. 4.3 is that the gait with AFOs does not allow for the ankle angle to be negative since it reduces the plantarflexion. Moreover, in barefoot walking scenario, the ankle angle assumes negative values, starting at -5° and reaching -10°.

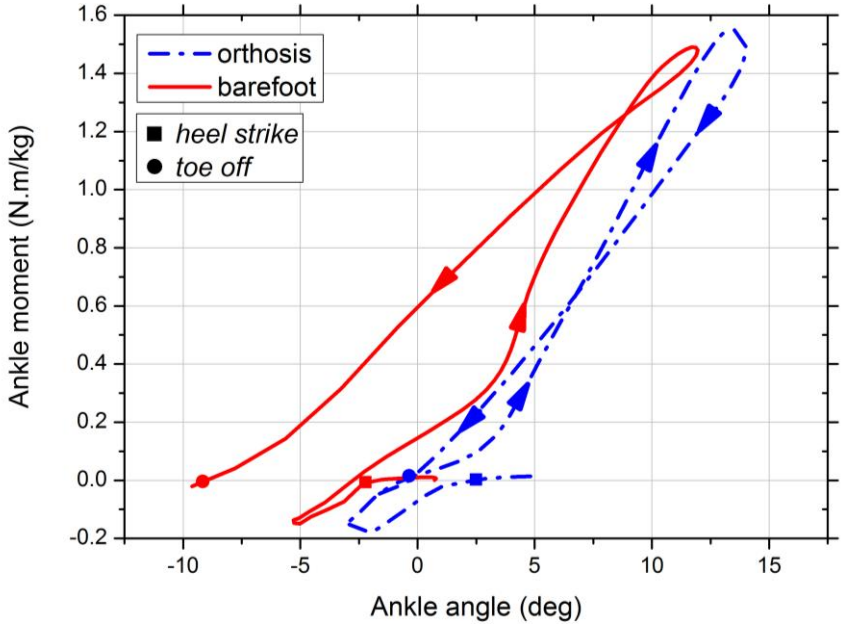


Figure 4.3 Moment/angle plots obtained in the gait lab **(a)** in a barefoot acquisition (red continuous line) and **(b)** with orthoses on both feet (blue dotted line)

4.1.2 Issues with the implementation

An application intended to reproduce the kinematic data captured in the gait lab was implemented in MOBILE. In this program, called "Reader", all the variables were rheonomic and the time functions were read from a *.dat* file. The first attempt consisted of applying directly the time function obtained in the gait lab, however some difficulties appeared and had to be fixed.

VICON® Nexus calculates the center of rotation of each joint. These points do not correspond to the markers positions because the center of rotation of a joint is not on the skin surface but inside the joint. This is particularly easy to understand for spherical joints like hip or shoulder. The hip was driven with 3 DOFs (2 translations and 1 rotation in the sagittal plane) with the data obtain for the PELO point, as Fig. 4.4 shows. The PELO point is a virtual point determined by VICON® Nexus that represents the middle point between the right (RHJC) and left (LHJC) hip joints centers of rotation (see Fig. 4.4).

VICON® Nexus calculates the joints angles automatically and these angles were extracted and applied directly at the joints in the simulation. However, the simulation using the angles obtained directly from VICON® Nexus revealed a non-natural walking with the feet going down and penetrating the floor during the stance phase, like if the model was walking on mud or snow. This issue has been reported on previous research as kinematic data inconsistency (Ceccarelli, 2009). The explanation for this problem is

that VICON® Nexus calculates the length of each segment frame-by-frame, based on the markers position. The problem is that the lengths change during the gait analysis because the markers are placed on the patient skin that is elastic and on the other hands the skin motion changes the lengths and affects the joint angles. As an example, during stance phase, the skin of the lower limbs is compressed and the calculated segments by VICON® Nexus become shorter. However, in MOBILE, only rigid bodies, which have a constant length, were used to build the MBS. To eliminate the kinematic inconsistencies, the average length of the segments in the sagittal plane were calculated and applied to the model, yet the abnormal gait pattern remained. As this solution did not solve completely the problem, it was necessary to compute the joint angles in MATLAB. The trigonometric relationships used for this purpose are described below. As the model is 2D in the sagittal-plane and the measurements are 3D, it was necessary to project the segments onto the vertical plane *yOz*. Though, as the subjects walked in the gait lab parallel to the *y*-axis, removal of the *x*-coordinate is enough to have the 2D-coordinates of the points (see Fig. 4.5).

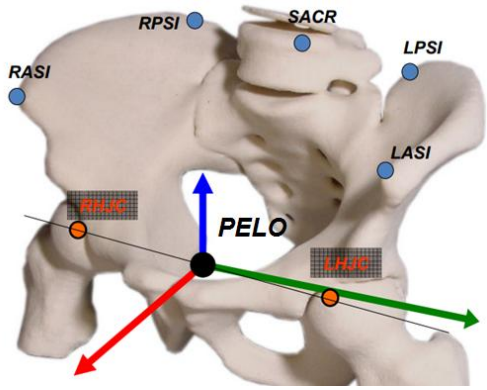


Figure 4.4 Position of the PELO point and the markers placed at the hip, adapted from Paolini (2010)

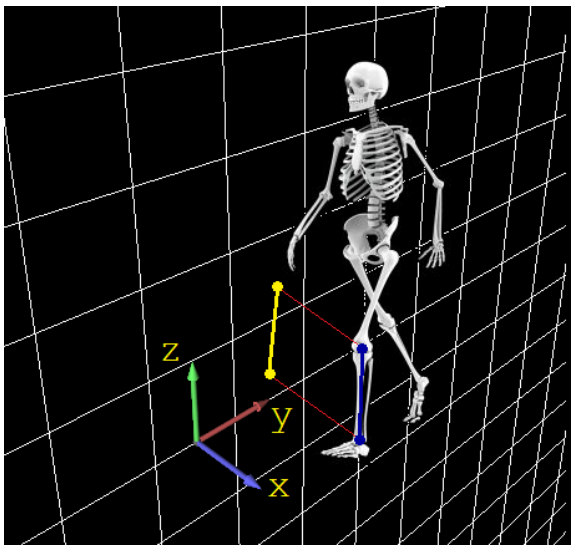


Figure 4.5 Projection of the coordinates in the *yOz* plane by removing the *x*-coordinate

Another important aspect is related to the points used for the angle computation. Whenever possible, the virtual points representing the center of rotation of the joints were used (PELO for the hip, R(L)FEO for the knees and R(L)TIO for the ankles) but some markers had to be used at the foot (RHEE, RTOE and R(L)TOETIP) (see Fig. 4.6).

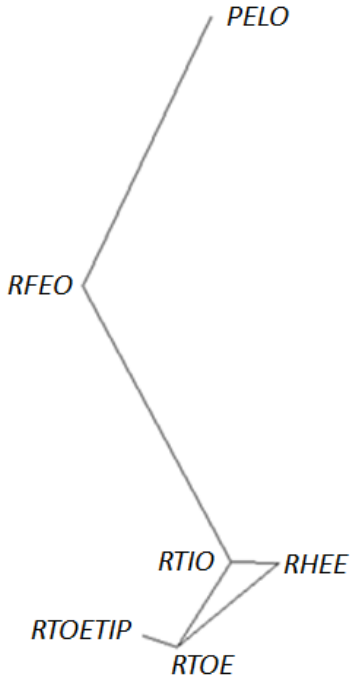


Figure 4.6 Identification of the points and markers used for the right lower limb angles computation

The hip was driven with 3 DOFs but the rotation was assumed to be always null during the simulation. This simplification was necessary because in this way, the other angles can be obtained by calculating the angles between the markers or virtual points.

4.1.2.1. Hip and knee angles

The hip and knee angles were obtained using simple trigonometric relations. The hip angle (α) is defined as the angle developed between the thigh and the z-axis (see Fig. 4.7). Using only a tangent relation, it is possible to write:

$$\alpha = arctan\left(\frac{RFEO_x - PELO_x}{RFEO_y - PELO_y}\right) \tag{4.1}$$

The knee angle (φ) is defined as the angle between the thigh and the shank (see Fig. 4.8) and is expressed by:

$$\varphi = \alpha + \beta, \text{ where } \beta = arctan\left(\frac{RTIO_x - RFEO_x}{RTIO_y - RFEO_y}\right) \tag{4.2}$$

4.1.2.2. Ankle angle

The ankle angle was the most complex angle to obtain. It is defined as the angle (δ) between the normal to foot sole and the shank (see Fig. 4.9). In order to compute it, two triangles were created, as illustrated in Fig. 4.9.

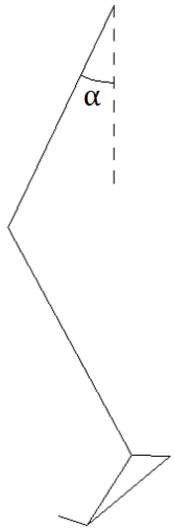


Figure 4.7 Hip angle (α)

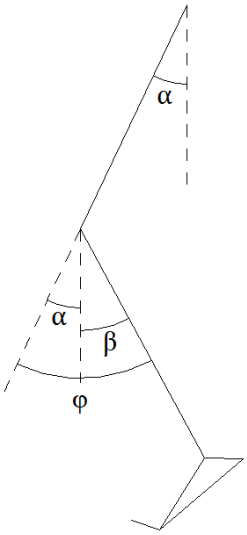


Figure 4.8 Knee angle (φ)

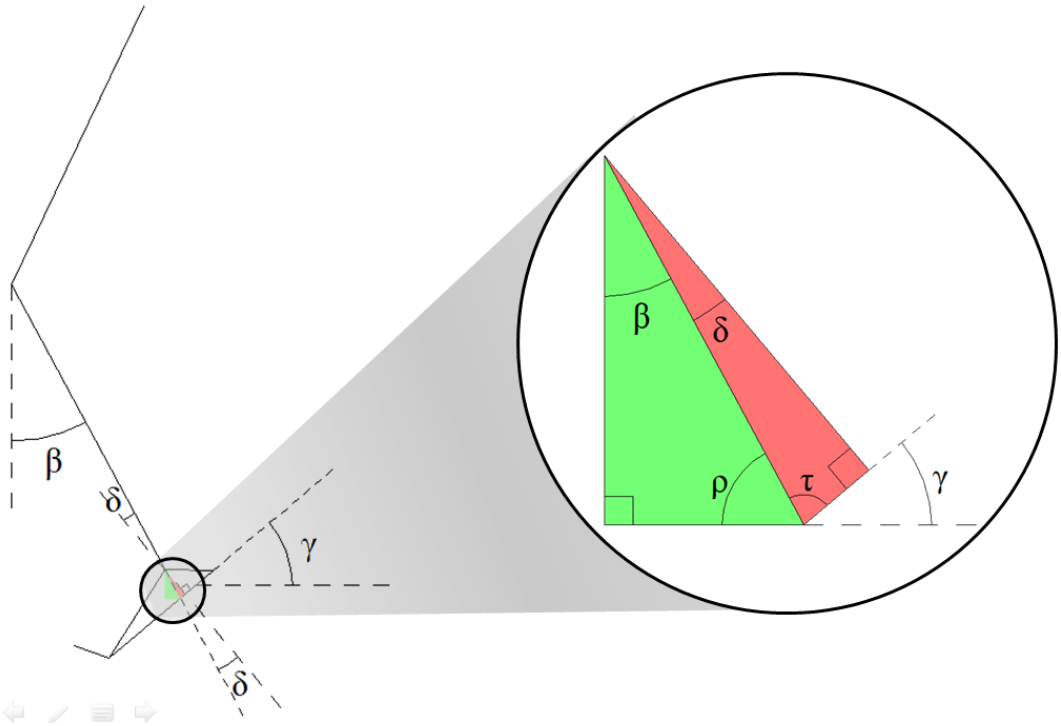


Figure 4.9 Ankle angle (δ)

By observing Fig. 4.9 reads

$$\begin{cases} \tau = 180 - \rho - \gamma \\ \rho = 90 - \beta \\ \delta = 90 - \tau \end{cases} \leftrightarrow \begin{cases} \tau = 90 + \beta - \gamma \end{cases} \leftrightarrow \begin{cases} \tau = 90 + \beta - \gamma \\ \delta = \gamma - \beta \end{cases}$$

Then, the ankle angle is obtained by the following equation.

$$\delta = \gamma - \beta, \text{ where } \gamma = arctan(\frac{RHEE_x - RTOE_x}{RHEE_y - RTOE_y}) \quad (4.3)$$

4.1.2.3. Metatarsal angle

The metatarsal angle (see Fig. 4.10) is the angle created by the articulation between the foot and the toes (ψ). It can be calculated by:

$$\psi = \varepsilon + \gamma, \text{ where } \gamma = arctan(\frac{RHEE_x - RTOE_x}{RHEE_y - RTOE_y}) \quad (4.4)$$

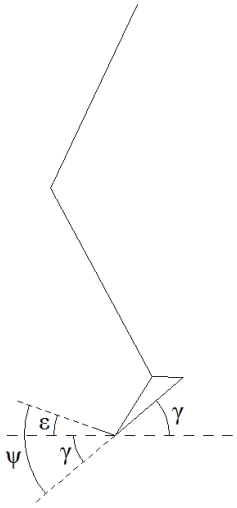


Figure 4.10 Metatarsal angle (ψ)

It must be noticed that angle ψ must be computed as $\psi = \varepsilon - \gamma$ because γ is negative.

4.1.2.4. HAT angle

According to Winter (2009), the HAT body can be defined as the body that connects the PELO point to the gleno-humeral joint (see Tab. 3.1). Similarly to the hip angle, the HAT angle (υ) is defined as the angle developed between the HAT and the vertical vector and can be computed as:

$$\upsilon = arctan(\frac{gleno\_humeral\_joint\ _x - PELO_x}{gleno\_humeral\_joint\ _y - PELO_y}) \quad (4.5)$$

In Equation (4.5), the gleno-humeral joint position was computed as the middle point between the shoulder centers of rotation (RHUP and LHUP).

$$\begin{aligned} gleno\_humeral\_joint\ _x &= \frac{RHUP_x + LHUP_x}{2} \\ gleno\_humeral\_joint\ _y &= \frac{RHUP_y + LHUP_y}{2} \end{aligned} \quad (4.6)$$

4.2 Reader procedure

The time functions obtained through Equations (4.1, 4.2, 4.3, 4.4 and 4.5) were used to guide the joints. The MOBILE application intended to validate the time functions was called “Reader”, since all the joints were guided with time functions. The “Reader” was able to walk with a normal gait pattern and without penetrating the floor, thus validating the code.

There is a specific procedure required for using the “Reader”. This algorithm guarantee the computation of all time functions and all steps should be completed as followed:

1. Acquire kinematic data in the gait lab (with or without orthoses), according to the gait lab procedures. An additional marker must be place at the toe tip, as shown in Fig. 4.2.
2. MATLAB must be opened and the total mass of the subject must be defined (e.g. $\mathrm{Tmass} = 65$), in order to normalize the moments to N.m/kg.
3. Run the *anthropometric.m* script to compute the rigid links properties (lengths, masses, moment of inertia and location of the CM) .
4. Run the *readc3d.m* script to compute the time functions.
5. Open MOBILE and run the “Reader” code to perform the simulation;

The simulation can be used to visualize the gait acquired in the gait lab and determine relevant instants of the gait cycle such as HS and TO. Right foot HS will be needed for forward dynamics simulation (Section 4.4).

4.3 Ankle angle over time plots

The results that validate the full-body model, as well as the foot contact model and the methodologies used, are the ankle kinematics. The results obtained in the gait lab (inverse dynamics) are reproduced using forward dynamics approach (see Section 4.4). The ankle kinematics is presented by plotting the ankle angle over time. A typical example is illustrated in Fig. 4.11, and shows clearly the variation of the ankle angle during the gait cycle. The sub-phases of stance period (see Fig. 2.11) are easily identified.

The interpretation of the ankle angle variation plotted in Fig. 4.11 clearly demonstrates how the ankle articulation works during the gait cycle (see Tab. 2.1). A full gait cycle was highlighted and analyzed in Fig. 4.11. It is possible to identify the two main phases of the gait cycle: stance and swing. The stance period sub-phases are

also identified: loading response (blue), mid-stance (red), terminal stance (yellow) and pre-swing (green). The maximum and minimum values of the plot correspond to the major events of the gait cycle (HS, *o*TO, *o*HS and TO), as they are identified in Fig. 4.11.

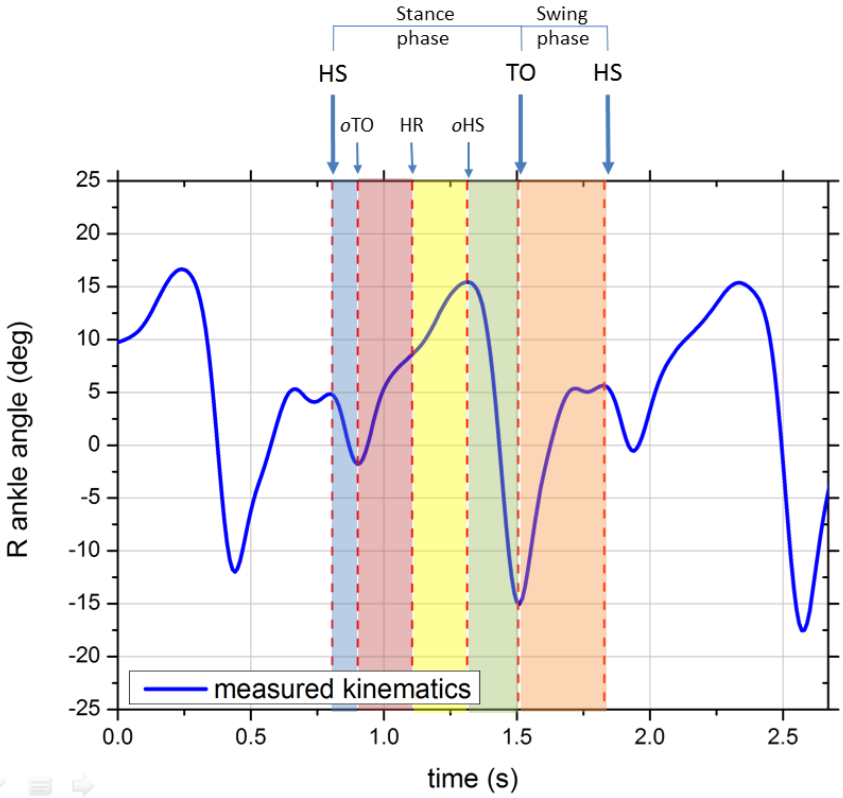
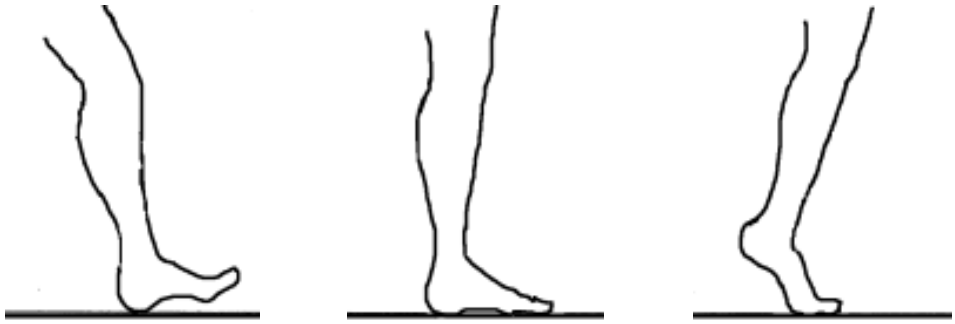


Figure 4.11 Typical plot obtained in the gait lab presenting the evolution of the ankle angle during the gait cycle. Heel strike (**HS**), toe off (**TO**), opposite TO (***o*TO**), heel rise (**HR**) and opposite HS (***o*HS**) were identified for a stride, as well as the stance and swing phase

During the loading response (blue), only the heel contacts the floor. At midstance (red), flat contact occurs since the whole foot sole is on the floor. During the terminal stance (yellow), the heel starts to rise and the peak of dorsiflexion is reached at opposite HS. Finally, during the pre-swing (green) the ankle makes a tremendous plantarflexion and pushes off the ground to propel the body forward and only the forefoot touches the floor. In order to better understand this idea, a graphical representation of the heel strike, foot flat and pre-wing is shown in Fig. 4.12a, 4.12b and 4.12c, respectively.

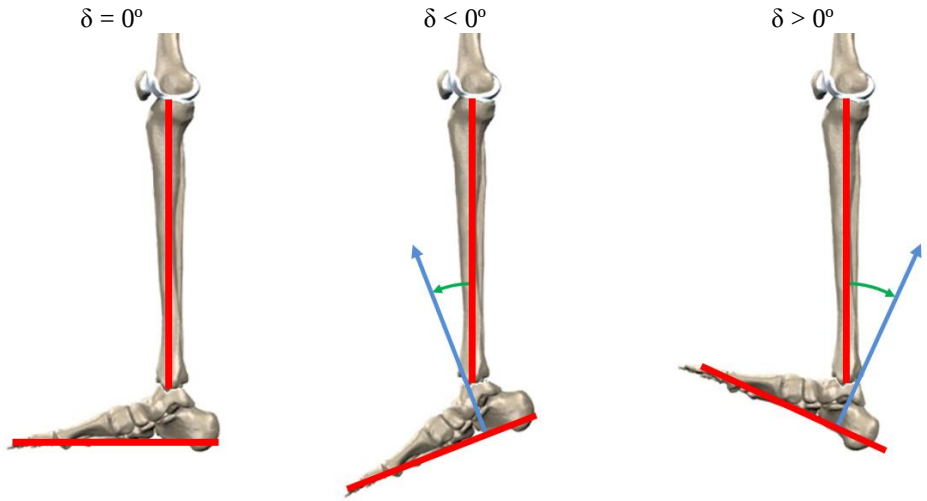
It must be noticed that the ankle angle is measured between the shank segment and the normal vector to the foot sole (see Fig. 4.9). Thus, the ankle angle is considered to be null when the shank and the foot sole are perpendicular to each other (see Fig. 4.13a). Since the angles are measured in the clockwise direction, the plantarflexion is

considered to be a movement in the negative direction (see Fig. 4.13b) and similarly, the dorsiflexion is positive (see Fig. 4.13c).



(a) **(b)** **(c)**

Figure 4.12 Graphical representation of some of the stance phase events: **(a)** heel strike, **(b)** foot flat and **(c)** pre-swing period, adapted from Voglewede (2007)



(a) **(b)** **(c)**

Figure 4.13 Representation of the ankle angle in three distinct configurations: **(a)** foot sole perpendicular to the shank, **(b)** foot in a plantarflexion position and **(c)** foot in a dorsiflexion position

4.4 Forward dynamics

The main goal of this section is to reproduce the data obtained in the gait lab in MOBILE, using forward dynamics. In this context, a set of three sphere-plane contacts, distributed at heel, metatarsal joint and toe tip, respectively, with a Hunt-Crossley's contact model (Hunt and Crossley, 1975) for computation of the contact forces, is considered (see Section 3.3.3). The MOBILE application created for this purpose is called "Integrator" and it differs from the "Reader" by having a LSodar integrator. LSodar is a MOBILE class used to integrate the variables capable to compute the variables and solve the system in forward dynamics.

4.4.1. Barefoot results

The first goal to test forward dynamics was to reproduce the ankle kinematics for the barefoot situation. In this context, the simulation process consists of identifying the right foot HS using the "Reader" application (see Section 4.2). Then, the "Integrator" is used and a set of joints of the MBS model (the pelvis joints, the metatarsal joints and the ankle joint of the stance foot) switches from rheonomic to scleronomic. By other words, these are now treated as generalized coordinates moving under the effect of forces according to the equations of motion. The remaining joints (knee and hip) still develop according to the time functions of the measured motion. Thus, the problem corresponds to a MBS with six DOFs subjected to six rheonomic constraints (2 hip joints, 2 knee joints, 1 ankle joint for the swing leg + 1 upper body joint). Only one of the ankle joints was set scleronomic because VICON® Nexus can only compute the ankle moment for the foot that hits the forces plates. In summation, the variables and their type (rheonomic or scleronomic) in both simulation conditions ("Reader" and "Integrator") are listed in Tab. 4.1.

Table 4.1 Comparison of the type of joints in the reader and integrator models

Joints	***Reader***	***Integrator***
HAT angle	rheonomic	rheonomic
Hip translation		
• *x*-direction	rheonomic	DOF
• *y*-direction	rheonomic	DOF
Hip rotation	rheonomic	DOF
Hip joints (both right and left)	rheonomic	rheonomic
Knee joints (both right and left)	rheonomic	rheonomic
Ankle joints		
• Right	rheonomic	DOF*
• Left	rheonomic	rheonomic
Metatarsal joints (both right and left)	rheonomic	DOF**

* The ankle moment obtained by inverse dynamics in the gait lab was applied to the right ankle joint.
** A spring-damper was added to the metatarsal joints.

The ankle moment computed by VICON Nexus® for the stance foot is applied directly to the ankle joint of the model in order to simulate the moment generated by the muscles. A typical example of the ankle moment during the measurements obtained by inverse dynamics is presented in Fig. 4.14. It must be noticed that the ankle moment is computed during a small window of time. This short time frame corresponds to the

period the right foot hits the force plates at the center of the gait lab, during the measurements.

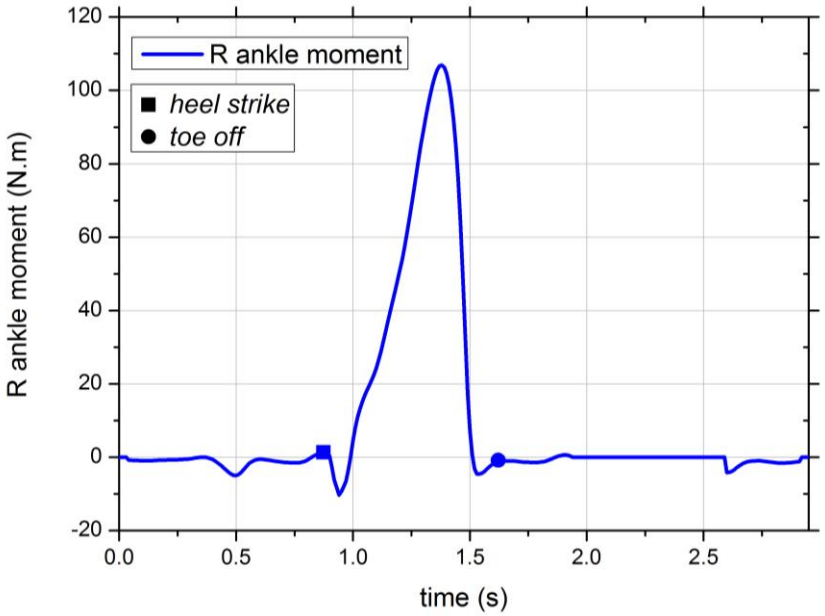


Figure 4.14 Graph presenting the evolution of the ankle moment during the measured gait

In order to simplify the scripts and the MOBILE code and standardize the results, it was defined that only the measurements were the right foot hit the force plate should be considered. During the acquisition of the kinematic data, the subject started to walk in such a manner that the force plate was always hit by the right foot.

The metatarsal joints time functions were also removed, and a spring-damper was applied at this joints. In this way, the foot model is able to adapt itself more easily and has a better behavior in contact to the floor when compared with the driven joints.

4.4.1.1 Optimization of the contact parameters

With this approach, it was necessary to obtain the contact and friction properties that reproduce the foot-floor interaction in a realistic manner. The optimized parameters were obtained by a trial-and-error procedure. The contact parameters were adjusted and, for each modification, the ankle kinematics were plotted and compared to the ankle kinematics acquired in the gait lab. The process was repeated until the results were considered satisfactory.

For simplification purposes, it was assumed that the metatarsal and toe tip spheres had the same contact properties. Therefore, only two different set of contact properties were optimized. This process was repeated for three different measurements and the

best results are the one discussed below. The corresponding set of parameters is presented in Tab. 4.2.

Table 4.2 Contact and friction properties that better reproduce the ankle joint kinematics

Contact parameters*	Heel sphere	Metatarsal and toe tip sphere
Normal stiffness c_N (N.m$^{-1.5}$)	6.5x10^5	1.0x10^9
Tangential stiffness c_T (N.m$^{-1.5}$)	6.5x10^5	1.0x10^9
Normal coefficient of restitution e_N	0.4	0.2
Tangential coefficient of restitution e_T	0.01	0.1
Exponent in the force-deflection function	1.5	1.5
Static/sticking friction coefficient μ_{st}	0.5	0.9
Dynamic/static friction coefficient μ_{sl}	0.4	0.8

* The contact parameters correspond to the parameters used by the MoRegImpSpherePlane class.

4.4.1.2 Optimization of the spring-damper constants

The metatarsal spring-damper parameters were obtained by the same procedure as the contact parameters (see Section 4.4.1.1). Actually, the optimization of both contact (see Tab. 4.2) and spring-damper (see Tab. 4.3) parameters occurred simultaneously to obtain a better compromise between the contact and the spring-damper parameters. It was assumed that the metatarsals joint behaved like a perfect spring without damping. The metatarsal spring and damper constants that better reproduce the ankle kinematics are reported in Tab. 4.3.

Table 4.3 Metatarsal spring-damper parameters

Spring-damper parameters	Value
Spring constant *k*	22.5 N.m/rad
Damper constant *c*	0.0 N.s/m

4.4.1.3 Barefoot results

Finally, the set of parameters and methodologies described in the previous sections were used to compute the ankle kinematics by forward dynamics. The results are presented in Fig. 4.15.

The plot presented in Fig. 4.15 shows that the model is able to correctly perform forward dynamics, since the ankle kinematics allows a good approximation of the ankle kinematics to the measured gait. These results refer to the right ankle angle from heel strike to toe off. It is not possible to reproduce the ankle angle during the swing phase because VICON only computes the moment applied at the ankle joint during the stance

phase. There is a small difference between the two plots, which can be explained by the time the model takes to stabilize after the heel strike. There is also a small delay between the measured and the forward dynamics graph, which starts at the beginning of the propulsion phase (of about 0.5s). However this delay is not too relevant since its maximum is only 0.04s. The last difference between the two plots is that, by forward dynamics, toe off is reached 0.04s sooner and with an angle 2.3 degrees smaller (in absolute value). However, the forward dynamics simulation shows a good correlation with the measured gait and globally, it is a clean curve with the same behavior as the one obtained by inverse dynamics.

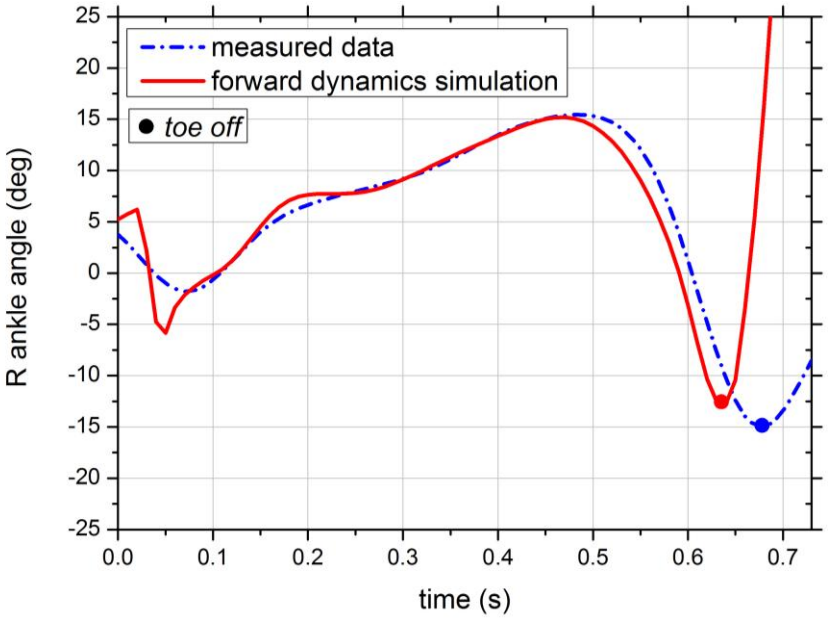


Figure 4.15 Ankle kinematics. **(a)** Measured (blue dotted line) and **(b)** obtained by forward dynamics (red line). These results were obtained for the barefoot trial with the best fit

4.4.2. Orthosis results

4.4.2.1. Ankle foot orthosis simulation

The first attempt to determine the AFOs properties was modeling the AFO in AutoCAD and then importing it in Ansys® (see Fig. 4.16). The idea behind this approach was to perform a dynamic analysis to the virtual model and adjust the properties of the orthosis so the deformation of the virtual model matches the experimental results get on mechanical assays. In this way, it would be possible to obtain the AFO properties and apply them to the MBS model. However, this idea was rejected because it was decided that the AFO would be added to the MOBILE model as

a spring-damper applied at the ankle joint and the spring and damper constants should be calculated in a different way (see Section 4.4.2.2).

Figure 4.16 Ankle-foot orthosis meshing in Ansys®

4.4.2.2. Orthosis simulation and results

After the biomechanical model (see Fig. 3.3) had proved to be reliable with the barefoot case (see Fig. 4.15), the code was used to reproduce the kinematic results measured in the gait lab with orthoses on both feet. The same methodologies as the one described in Section 4.4.1 were used, trial-and-error, which includes a few differences. Thus, a spring-damper system was added at the right ankle joint, thus replacing the AFO. In addition, the moment applied at this ankle joint was multiplied by a coefficient α (with $0 < \alpha < 1$) in order to mimic the reduction in muscular activation produced by the orthosis. In other words, α represents the percentage of muscular activation needed to maintain a normal gait using an AFO, comparing to the barefoot case. The foot parameters were the same as used in the barefoot case (see Tabs. 4.2 and 4.3) since patients use normally the same kind of footwear whether they use the AFO or not.

Moreover, the rheonomic/scleronomic constraints were the same as the one presented in Tab. 4.1 and the time function used in the rheonomic constraints corresponded to the gait measured experimentally without orthosis. The goal of this simulation was to reproduce the ankle kinematics acquired in the gait lab with an orthosis using kinematics obtained barefoot. Thereby, it is possible to assess the AFO effect on the non-pathological gait and calculate the AFO properties.

The optimization process occurred by trial-and-error by tuning the AFO spring-damper properties and the coefficient α, which are presented in Tab. 4.4 and Fig. 4.17, respectively. Based on the results showed in Fig. 4.3, it was assumed that there was no

damping, and the AFO should only be modeled as a linear torsional spring. Therefore, the damper constant was set null (see Fig. 4.4).

Table 4.4 AFO spring-damper parameters

Spring-damper parameters	Value
Spring constant *k*	50.0 N.m/rad
Damper constant *c*	0.0 N.s/m

By analysis Tab. 4.4, it is possible to assess the AFO is comparable to a spring with a spring constant equal to 50.0 N.m/rad.

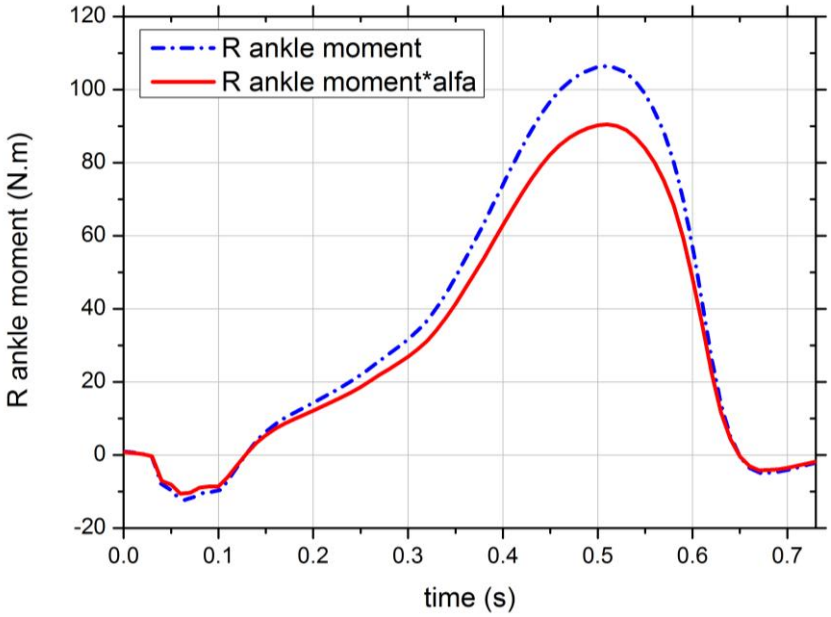


Figure 4.17 Moment applied at the ankle for the **(a)** barefoot simulation (blue dotted line) and **(b)** the simulation with AFO with α = 0.85 (red continuous line)

As it was previously mentioned, the ankle moment was multiplied by a coefficient α (see Fig. 4.17) and the ankle kinematics closest to the one measured with an AFO (see Fig. 4.17) were obtained for α equal to 0.85. That means that when using an AFO with $k = 50\ N.m/rad$, the muscular activation at the ankle joint is only 85% of the muscular activation of the same gait barefoot. Thus, it can be concluded that the orthosis reduces the muscular activation at the ankle joint in 15%, for a non-pathological subject. Similarly to the barefoot case, the ankle kinematics with orthoses obtained by forward dynamics were plotted against time and compared it with the measurements captured in the gait lab to validate the model (see Fig. 4.18).

Ankle kinematics presented in Fig. 4.18 refers to the right ankle angle evolution during stance phase. Only a few differences between the measured gait and the simulation result can be identified during the stance phase. The plot obtained by

forward dynamics is considered satisfactory since it matches quite well to the one measured in the gait lab.

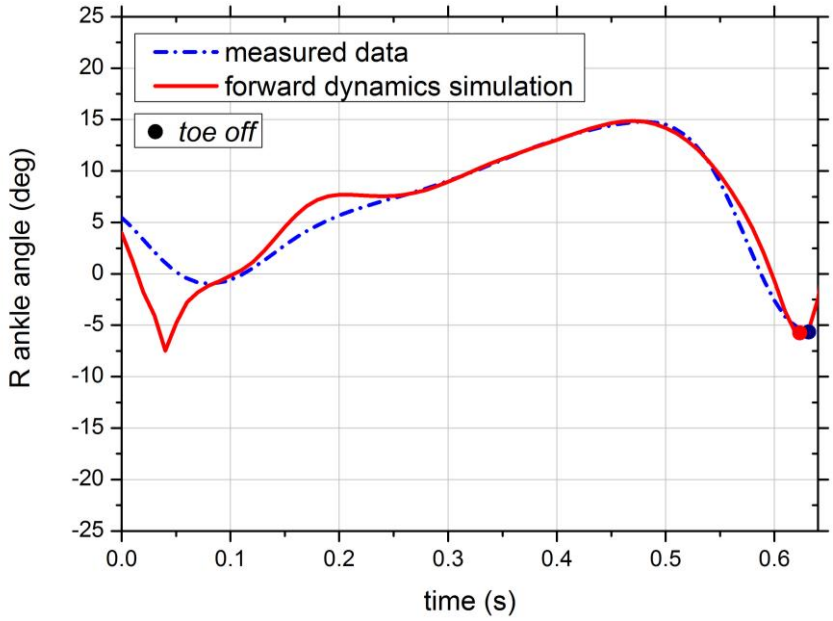


Figure 4.18 Ankle kinematics. **(a)** Measured (blue dotted line) and **(b)** obtained by forward dynamics (red line). These results were obtained for the orthosis trial with the best fit

Equally to the barefoot simulation (see Fig. 4.15), the initial instant does not exhibit a good solution because of the heel strike impact and the simulation takes a little time to stabilize. The difference between the diagrams reaches 6 degrees but after approximately 0.1s, the forward dynamics curve approaches the curve obtained in the measured gait. Thus, the model was validated since it allows a good correlation to the ankle kinematics measured in the gait lab, during the stance phase.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Ankle foot orthoses

AFOs are a suitable solution for patients suffering from a great variety of pathologies, such as cerebral palsy, arthritis and nerve damage. They are also indicated for patients suffering from a symptom called dropfoot that is characterized by an excessive plantarflexion and the inhability to dorsiflex the foot. Usually, patients with dropfoot drag their feet on the floor during the swing phase causing pains and injuries. In these cases, patients generally over-compensate the dropfoot drag by increasing and adding a lateral hip moment. AFOs prevent the foot from dragging in the floor by supporting it and blocking plantarflexion. Usually, patients gait pattern approaches the normal gait and they recover the ability to walk by themselves.

The biomechanical evaluation of the AFOs effect assesses that this type of orthosis improve even the non-pathological gait. With an AFO acting on the ankle joint, there is no damping, so the joint acts like a nearly ideal spring.

5.2 MOBILE and methodologies

A 2D multimodel of the human body in the sagittal plane was developed in MOBILE software. The model was created for the sagittal plane only with by 9 rigid bodies constrained by 12 frictionless joints (11 revolute and 1 prismatic joints) that can be driven with time functions.

The full-body model is intended to solve forward dynamics problems and the project goal was achieved since the ankle kinematics obtained in the gait lab was positively reproduced during stance phase, with and without orthosis.

A simple contact model was developed in the foot-floor interface with three sphere-plane contacts. This model proved to be suitable for this study because it is efficient and simple to define. Results were considered satisfactory and the model was validated.

An AFO can be simulated as a torsional spring acting at the ankle joint since the ankle moment varies linearly with the ankle angle during the stance phase when using this type of orthosis. Kinematic measurements also demonstrated that the orthopedic device fulfills its function by restricting plantarflexion, which was proved by smaller range of plantarflexion when wearing the orthosis.

Forward dynamics simulation proved that the AFO used in this experiment is equivalent to an ideal torsional spring with $k = 50.0\ \mathrm{N.m/rad}$. For a healthy subject,

using an AFO reduces the muscular activity at the ankle joint in about 15%, compared to the gait without orthoses.

5.3 Advantages of the methodologies

MOBILE was the chosen programing environment and it proved to be an appropriate choice, since it is designed for the modeling of MBS. It is implemented in C++ language and it is necessary to possess the basic concepts to deal with it but, after a while, the structure and the basic objects became simple to understand. MOBILE proved to be a software prepared to read kinematic data obtained in a gait lab and to reproduce them, using forward dynamics.

All the data captured in the gait lab were already synchronized. This is a main advantage because it eliminates the issues associated with synchronization and the data are immediately ready to be read by the MATLAB and MOBILE scripts.

5.4 Limitations

Globally, results computed using the MBS model fitted quit well the measured values. However, as the model is 2D, movement in the frontal and horizontal planes are ignored. Thus, the model cannot be used to perfectly replicate how the human body works, since the human joints move in 3D. Nevertheless, it can give a good approximation to the sagittal plane movement but the kinematic data must be acquired carefully in the gait lab with the subject walking parallel to the walkway.

It is not possible to apply this method to compare the gait of a patient with and without orthoses because normally patients over-compensate the dropfoot pathologies with a lateral elevation of the leg at the hip. This moment created at the hip does not fit in the sagittal plane and it is significantly reduce with the use of the orthoses. In this way, it is still not known how to predict the effect of orthoses on those patients kinematics. However the procedure employed in this work can be used with disable people to compare the effectiveness of different orthoses.

5.5 Future work

In general, the results (see Figs. 4.15 and 4.18) were obtained by a manual iterative method of trial-and-error. They can be improved with an optimizer in order to reduce the discrepancy between the forward dynamics results from the measured ones. The MOBILE optimizer could not be applied as part of this work because a problem

with the optimizer class only allows the simulation to run once, and it must be closed and restarted each time.

The contact foot model used was very simple, since it has only three contact points. Using a more complex and accurate contact model which includes midfoot contact may be reflected in better results.

REFERENCES

ABBOUD, R. J. 2002. Relevant foot biomechanics. *Current Orthopaedics,* 16**,** 165-179.

ADVANCED ORTHOTIC DESIGNS INC. 2011. *Advanced Orthotic Design - Foot* [Online]. Mississauga: Advanced Orthotic Designs Inc. Available: http://aodmobility.com/custom-devices/foot/ [Accessed 21-09-2011].

ALEXANDER, M. A., XING, S. Y. & BHAGIA, S. M. 2011. *Lower limb orthotics* [Online]. WebMD LLC. Available: http://emedicine.medscape.com/article/314838-overview#aw2aab6b5 [Accessed 22-09-2011].

AMERICAN BOARD FOR CERTIfiCATION IN ORTHOTICS PROSTHETICS AND PEDORTHICS INC. 2007. Practice analysis of certified practitioners in the disciplines of orthotics and prosthetics. Alexandria, Virginia, USA.

AMERICAN ORTHOTIC AND PROSTHETIC ASSOCIATION INC. 2008. Evidence note - the use of ankle-foot orthoses in the management of stroke.

BREGMAN, D. J. J., ROZUMALSKI, A., KOOPS, D., DE GROOT, V., SCHWARTZ, M. & HARLAAR, J. 2009. A new method for evaluating ankle-foot orthosis characteristics: BRUCE. *Gait & Posture,* 30**,** 144-149.

BREGMAN, D. J. J., VAN DER KROGT, M. M., DE GROOT, V., HARLAAR, J., WISSE, M. & COLLINS, S. H. 2011. The effect of ankle foot orthosis stiffness on the energy cost of walking: a simulation study. *Clinical Biomechanics (Bristol, Avon),* 26**,** 955-961.

BREHM, M.-A., HARLAAR, J. & SCHWARTZ, M. 2008. Effect of ankle-foot orthoses on walking efficiency and gait in children with cerebral palsy. *Journal of Rehabilitation Medicine,* 40**,** 529-534.

CALLAGHAN, M. J. & BALTZOPOULOS, V. 1994. Gait analysis in patients with anterior knee pain. *Clinical Biomechanics,* 9**,** 79-84.

CECCARELLI, M. 2009. *Proceedings of EUCOMES 08: the second european conference on Mechanism Science,* Cassino, Italy, Springer.

CHEN, C.-L., YEUNG, K.-T., WANG, C.-H., CHU, H.-T. & YEH, C.-Y. 1999. Anterior ankle-foot orthosis effects on postural stability in hemiplegic patients. *Archives of Physical Medicine and Rehabilitation,* 80**,** 1587-1592.

CHU, T. M. 2000. Determination of peak stress on PP AFOs due to weight change using strain gage technology. *Experimental Techniques,* 24**,** 28-30.

CHU, T. M. 2001. Biomechanics of ankle-foot orthoses: past, present, and future. *Topics in Stroke Rehabilitation,* 7**,** 19-28.

CHU, T. M. & FENG, R. 1998. Determination of stress distribution in various ankle-foot orthoses: experimental stress analysis. *JPO: Journal of Prosthetics and Orthotics,* 10**,** 11-16.

CHU, T. M. & REDDY, N. P. 1995. Stress distribution in the ankle-foot orthosis used to correct pathological gait. *Journal of Rehabilitation Research and Development,* 32**,** 349-60.

CHU, T. M., REDDY, N. P. & PADOVAN, J. 1995. Three-dimensional finite element stress analysis of the polypropylene, ankle-foot orthosis: static analysis. *Medical Engineering & Physics,* 17**,** 372-379.

COOPER, G. 2006. *Essential physical medicine and rehabilitation,* New Jersey, USA, Humana Press.

COUTINHO, J., SILVA, P. & FIGUEIREDO-PINA, C. Ensaios de aplicação localizada de força numa perna no limiar da dor. 4º Congresso Nacional de Biomecânica, 04-05 Feb. 2011 Coimbra, Portugal. 623-626.

CRABTREE, C. A. & HIGGINSON, J. S. 2009. Modeling neuromuscular effects of ankle foot orthoses (AFOs) in computer simulations of gait. *Gait & Posture,* 29**,** 65-70.

DE BURGH, J. 2003. *The human body - an essential guide to how the body works,* Rochester, U.K., Grange Books.

EDELSTEIN, J. E. & BRUCKNER, J. 2002. *Orthotics: a comprehensive clinical approach,* New Jersey, USA, Slack.

FLORES, P., AMBRÓSIO, J., CLARO, J. C. P. & LANKARANI, H. M. 2006. Influence of the contact–impact force model on the dynamic response of multi-body systems. *Proceedings of the Institution of Mechanical Engineers. Part K : Journal of Multi-body Dynamics,* 220**,** 21-34.

GOLAY, W., LUNSFORD, T., LUNSFORD, B. R. & GREENFIELD, J. 1989. The effect of malleolar prominence on polypropylene AFO rigidity and buckling. *JPO: Journal of Prosthetics and Orthotics,* 1**,** 231-241.

GORDON, K. E., SAWICKI, G. S. & FERRIS, D. P. 2006. Mechanical performance of artificial pneumatic muscles to power an ankle-foot orthosis. *Journal of Biomechanics,* 39**,** 1832-1841.

GRABNER, G. 2003. *Efficient and reliable multibody simulation of regularized impacts between elementary contact pairs.* Ph.D., Technische Universitaet Graz, Austria.

HANSEN, A. H., CHILDRESS, D. S., MIFF, S. C., GARD, S. A. & MESPLAY, K. P. 2004. The human ankle during walking: implications for design of biomimetic ankle prostheses. *Journal of Biomechanics,* 37**,** 1467-1474.

HARRIS, G. F., SMITH, P. A. & MARKS, R. M. 2008. *Foot and ankle motion analysis: clinical treatment and technology,* Boca Raton, USA, CRC Press.

HUANG, Y.-C., HARBST, K., KOTAJARVI, B., HANSEN, D., KOFF, M. F., KITAOKA, H. B. & KAUFMAN, K. R. 2006. Effects of ankle-foot orthoses on ankle and foot kinematics in patients with subtalar osteoarthritis. *Archives of Physical Medicine and Rehabilitation,* 87**,** 1131-1136.

HUNT, K. H. & CROSSLEY, F. R. E. 1975. Coefficient of restitution interpreted as damping in vibroimpact. *Journal of Applied Mechanics,* 42**,** 440-445.

INTERNATIONAL SOCIETY OF BIOMECHANICS. 2012. *Signal processing software* [Online]. Available: http://isbweb.org/software/sigproc.html [Accessed 17-10-2012].

JAMSHIDI, N., ROSTAMI, M., NAJARIAN, S., MENHAJ, M. B., SAADATNIA, M. & FARZAD, A. Gait modeling for assessment of ankle-foot orthosis.

Biomedical Engineering Conference, 2008. CIBEC 2008. Cairo International, 18-20 Dec. 2008. 1-4.

JAMSHIDI, N., ROSTAMI, M., NAJARIAN, S., MENHAJ, M. B., SAADATNIA, M. & FIROOZ, S. 2009. Modelling of human walking to optimise the function of ankle-foot orthosis in Guillan-Barré patients with drop foot. *Singapore Medical Journal,* 50(4)**,** 412-417.

KECSKEMÉTHY, A. 1999. *MOBILE1.3, user's guide*, Duisburg, Germany, Ingenieurwissenschaften: Lehrstuhl für Mechanik und Robotik, Universität Duisburg-Essen.

KECSKEMÉTHY, A. A novel cylinder-plane foot contact model for human gait motion reproduction. ECCOMAS Multibody Dynamics 2011, 04-07 Jul. 2011 Brussels, Belgium. 1-5.

KINETIC RESEARCH INC. 2011. *The noodle AFO carbon fiber brace* [Online]. Lutz, Florida, USA. Available: http://www.kineticr.com/noodle.html 22-09-2011]].

KNIT-RITE. 2010. *Knit-Rite SmartKnit® AFO Liner* [Online]. Kansas City. Available: http://www.knitrite.com/Products/Orthotic_Textiles/AFO_KAFO_Socks_Product/SmartKnit_AFO_Liner.html [Accessed 22-09-2011].

LAI, H.-J., YU, C.-H., KAO, H.-C., CHEN, W.-C., CHOU, C.-W. & CHENG, C.-K. 2010. Ankle-foot simulator development for testing ankle-foot orthoses. *Medical Engineering & Physics,* 32**,** 623-629.

LAMONTAGNE, A., MALOUIN, F., RICHARDS, C. L. & DUMAS, F. 2002. Mechanisms of disturbed motor control in ankle weakness during gait after stroke. *Gait & Posture,* 15**,** 244-255.

LEONE, D., DIEMENTE, S., GUSTAVE, S. & LOPEZ-ISA, M. Structural analysis of solid ankle-foot orthoses. Bioengineering Conference, 1988., Proceedings of the 1988 Fourteenth Annual Northeast, 10-11 Mar. 1988. 26-28.

LEONE, D., DIEMENTE, S. & LOPEZ-ISA, M. Structural stability prediction for thermoplastic ankle-foot orthoses. Bioengineering Conference, 1991., Proceedings of the 1991 IEEE Seventeenth Annual Northeast, 04-05 Apr. 1991. 231-232.

LOWER EXTREMITY REVIEW MAGAZINE. 2011. *EZ-stride floor reaction AFO* [Online]. Available: http://www.lowerextremityreview.com/products/ez-stride-floor-reaction-afo [Accessed 22-09-2011].

MILLARD, M., MCPHEE, J. & KUBICA, E. 2008. Multi-step forward dynamic gait simulation - multibody dynamics. *In:* BOTTASSO, C. L. (ed.). Springer Netherlands.

MOREIRA, P. 2009. *Development of a 3D contact model for the foot-ground interaction in gait simulations.* M.Sc., Universidade do Minho, Portugal.

MOREIRA, P., SILVA, M. & FLORES, P. Ground foot interaction in human gait: modelling and simulation. 7th EUROMECH Solid Mechanics Conference, 07-11 Sep. 2009 Lisbon, Portugal.

NOVITATECH. 2010. *Ankle foot orthoses (AFOs) - customised solutions* [Online]. Novita Available: http://www.novitatech.org.au/content.asp?p=414 [Accessed 22-09-2011].

ORTHOMEDICS. 2011. *AFO - ankle foot orthosis* [Online]. Omaha: OrthoMedics Orthotic and Prosthetic Services. Available: http://orthomedics.us/ankle.aspx [Accessed 22-09-2011].

PAKISTAN ACADEMY OF ORTHOTISTS & PROSTHETISTS. 2009. *Basic biomechanics of ankle foot orthoses (AFOs)* [Online]. Available: http://www.paop.pk/web%20books/Orthotics%20at%20OLC/4%20AFOBiomech.pdf [Accessed 13-09-2012].

PAOLINI, G. 2010. Interpreting PiG results: PiG biomechanical modelling. *Plug in Gait WebEx Training,* Session 3.

PROWALK GMBH. 2009. *Parawalker | ProWalk* [Online]. Egelsbach. Available: http://www.prowalk.de/Parawalker/450/ [Accessed 21-09-2011].

ROMKES, J. & BRUNNER, R. 2002. Comparison of a dynamic and a hinged ankle-foot orthosis by gait analysis in patients with hemiplegic cerebral palsy. *Gait & Posture,* 15**,** 18-24.

ROSE, J. & GAMBLE, J. G. 2005. *Human walking,* Philadelfia, USA, Lippincott Williams & Wilkins.

SILVA, P., SILVA, M. & MARTINS, J. 2010. Evaluation of the contact forces developed in the lower limb/orthosis interface for comfort design. *Multibody System Dynamics,* 24**,** 367-388.

SILVA, P., SILVA, M. T. & MARTINS, J. Alteração dos parâmetros de marcha devido à utilização de uma ortótese. 4º Congresso Nacional de Biomecânica, 04-05 Feb. 2011 Coimbra, Portugal. 633-637.

SUPAN, T. J. & HOVORKA, C. F. 1995. A review of thermoplastic ankle-foot orthoses adjustments/replacements in young cerebral palsy and spina bifida patients. *JPO: Journal of Prosthetics and Orthotics,* 7**,** 15-22.

TORTORA, G. J. & DERRICKSON, B. H. 2008. *Principles of anatomy and physiology,* New Jersey, USA, John Wiley & Sons.

VANKOSKI, S. J., SARWARK, J. F., MOORE, C. & DIAS, L. 1995. Characteristic pelvic, hip, and knee kinematic patterns in children with lumbosacral myelomeningocele. *Gait & Posture,* 3**,** 51-57.

VASCONSELOS, C. A. F. 2010. *Active Oorthosis for ankle articulation pathologies.* M.Sc., Universidade Técnica de Lisboa, Portugal.

VICON. 2012. *What are the details of the Woltring filter?* [Online]. Oxford, United Kingdom. Available: http://www.vicon.com/support/solution_view.php?id=1098 [Accessed 17-10-2012].

VOGLEWEDE, P. 2007. *Normal gait characteristics* [Online]. Department of Mechanical Engineering: University of South Carolina, USA. Available: http://www.me.sc.edu/fs/voglewede/design(2).htm [Accessed 15-09-2012].

WHITTLE, M. 2007. *Gait analysis: an introduction,* Philadelfia, USA, Butterworth-Heinemann Elsevier.

WINTER, D. A. 2009. *Biomechanics and motor control of human movement,* New Jersey, USA, John Wiley & Sons Inc.

WOLTRING, H. J. 1986. A Fortran package for generalized, cross-validatory spline smoothing and differentiation. *Advances in Engineering Software (1978),* 8**,** 104-113.

YAMAMOTO, S., EBINA, M., IWASAKI, M., KUBO, S., KAWAI, H. & HAYASHI, T. 1993. Comparative study of mechanical characteristics of plastic AFOs. *JPO: Journal of Prosthetics and Orthotics,* 5**,** 59/47-52/64.

APPENDIX I

N'PENDULUM CODE FOR MOBILE

```
#include <Mobile/MoBase.h>
#include <Mobile/MoMapChain.h>
#include <Mobile/MoElementaryJoint.h>
#include <Mobile/MoRigidLink.h>
#include <Mobile/MoMassElement.h>
#include <Mobile/MoMechanicalSystem.h>
#include <Mobile/MoAdamsIntegrator.h>
#include <Mobile/Inventor/MoScene.h>


int main () {

	int n; //number of bars
	cout << ''N' PENDULUM CREATOR' << '\n\n' << 'Enter the number of
bars: ';
	cin >> n;

	MoFrame * frames = new MoFrame[((2*n)+1)];
	MoAngularVariable * angles = new MoAngularVariable[n];
	MoVector * lenght = new MoVector[n];
	MoElementaryJoint * joints = new MoElementaryJoint[n];
	MoRigidLink * bars = new MoRigidLink[n];
	MoMassElement * mass = new MoMassElement[n];

	//create elementary joints
	for (int i=0, j=0; j<n; i=i+2, j++){
		//MoElementaryJoint temp_joint (frames[i], frames[i+1],
angles[j], yAxis);
		//joints[j] = temp_joint;
		joints[j].init(frames[i], frames[i+1], angles[j], yAxis);
	}

	//Lenght of the bars
	for (int i=0; i<n; i++){                              //first
clear all the values
		lenght[i] = MoNullState;
	}
	for (int i=0; i<n; i++){                              //then ask
for the lenght
		cout << 'Insert lenght of bar ' << i+1 << ': ';
		cin >> lenght[i].z;
	}

	//create rigid bars
	for (int i=0, j=0; j<n; i=i+2, j++){
		//MoRigidLink temp_link (frames[i+1], frames[i+2],
lenght[j]);
		//bars[j] = temp_link;
		bars[j].init(frames[i+1], frames[i+2], lenght[j]);
	}
```

```
      //Mass
      for (int i=0, j=2; i<n; i++, j=j+2){
            cout << 'Insert mass at the end of bar ' << i+1 << ': ';
            MoReal m;
            cin >> m;
            //MoMassElement temp_mass (frames[j], m);
            //mass[i] = temp_mass;
            mass[i].init(frames[j], m);
      }

      //Initial angles
      for (int i=0; i<n; i++){
            cout << 'Insert initial angle (in degrees) between bar ' <<
i << ' and ' << i+1 << ': ';
            int ang;
            cin >> ang;
            angles[i].q = DEG_TO_RAD * ang;
      }

      //Define the mechanical system
      MoMapChain Pendulum;
      for (int i=0; i<n; i++){                        //first add the
joints and the bars
            Pendulum << joints[i] << bars[i];
      }
      for (int i=0; i<n; i++){                        //then add the
masses
            Pendulum << mass[i];
      }

      //Create a list of generalized coordinates & dynamic equation
      MoVariableList vars;
      for (int i=0; i<n; i++){
            vars << angles[i];
      }

      MoVector gravity(0.0,0.0,-9.8);

      MoMechanicalSystem  Dynamics ( vars, Pendulum, frames[0],
gravity );

      //Numerical integrator
      MoAdamsIntegrator dynamicMotion (Dynamics);
      MoReal dT = 0.01;
      MoReal tol = 0.01;
      dynamicMotion.setTimeInterval(dT);
      dynamicMotion.setRelativeTolerance(tol);
```

```
	Pendulum.doMotion(DO_POSITION);

	//animation
	MoScene Scene (Pendulum);
	//creating revolute joints
	for (int i=0; i<n; i++){
		Scene.makeShape(joints[i]);
		Scene.makeShape (joints[i], bars[i]);
	}
	//Show frame at the origin
	Scene.makeShape ( frames[0] );

	Scene.addAnimationObject (dynamicMotion);
	Scene.setAnimationIncrement (dT);
	Scene.show();
	MoScene::mainLoop();

	return 0 ;
}
```

APPENDIX II

READER.CPP

```
#include <Mobile/MoBase.h>
#include <Mobile/MoMapChain.h>
#include <Mobile/MoElementaryJoint.h>
#include <Mobile/MoRigidLink.h>
#include <Mobile/MoMassElement.h>
#include <Mobile/MoMechanicalSystem.h>
#include <Mobile/MoAdamsIntegrator.h>

#include <MoUtilities/MoInterpolatingVarListInput.h>
#include <MoUtilities/MoVarListOutput.h>

#include <fstream>

#include 'MyClock.h'

#include <Mobile/Inventor/MoScene.h>
#include <Mobile/Inventor/MoWidget.h>

int main () {

/*********************************************************************
*                        DEFINITION OF ALL ELEMENTS                  *
*********************************************************************

/*    (1)   MODEL
*********************************************************************
//Definition of mechanical system
MoFrame K0, K1, K2, K3, K4, K5, K6, K7, K8, K9, K10, K11, K12, K13,
K14, K15, K16, K17, K18, K19, K20, K21, K22, K23, K24, K25, K26;
MoAngularVariable rotZ, ang1, ang2, ang3, ang4, ang5, ang6, ang7,
ang8, angHAT;
MoLinearVariable trX, trY;
MoVector l_virt1, l_virt2, l_virt3, l_HAT,  l_R_thigh, l_R_shank,
l_R_foot[2], l_R_toes, l_L_thigh, l_L_shank, l_L_foot[2], l_L_toes;
MoFrameList R_K_Outs, L_K_Outs;
R_K_Outs << K12 << K13;
L_K_Outs << K21 << K22;

//Joints
MoElementaryJoint xTrans      (K0, K1, trX, xAxis)    ;
MoElementaryJoint yTrans      (K2, K3, trY, yAxis)    ;
MoElementaryJoint Rot         (K4, K5, rotZ, zAxis)   ;
MoElementaryJoint R_hip       (K6, K7, ang1, zAxis)   ;
MoElementaryJoint R_knee      (K8, K9, ang2, zAxis)   ;
MoElementaryJoint R_ankle     (K10, K11, ang3, zAxis) ;
MoElementaryJoint R_toes      (K13, K14, ang4, zAxis) ;
MoElementaryJoint L_hip       (K6, K16, ang5, zAxis)  ;
MoElementaryJoint L_knee      (K17, K18, ang6, zAxis) ;
MoElementaryJoint L_ankle     (K19, K20, ang7, zAxis) ;
```

```
MoElementaryJoint L_toes      (K22, K23, ang8, zAxis) ;
MoElementaryJoint HAT_rot   (K6, K25, angHAT, zAxis)  ;

//Links
MoRigidLink virt1       (K1, K2, l_virt1) ;
MoRigidLink virt2       (K3, K4, l_virt2) ;
MoRigidLink virt3       (K5, K6, l_virt2) ;
MoRigidLink R_thigh     (K7, K8, l_R_thigh) ;
MoRigidLink R_shank     (K9, K10, l_R_shank) ;
MoRigidLink R_foot      (K11, R_K_Outs, l_R_foot) ;
MoRigidLink R_toe       (K14, K15, l_R_toes) ;
MoRigidLink L_thigh     (K16, K17, l_L_thigh) ;
MoRigidLink L_shank     (K18, K19, l_L_shank) ;
MoRigidLink L_foot      (K20, L_K_Outs, l_L_foot) ;
MoRigidLink L_toe       (K23, K24, l_L_toes) ;
MoRigidLink HAT         (K25, K26, l_HAT) ;

//Masses values and inertial tensors
MoReal m_HAT, m_R_thigh, m_R_shank, m_R_foot, m_R_toes, m_L_thigh,
m_L_shank, m_L_foot, m_L_toes;
MoInertiaTensor T_HAT, T_R_thigh, T_R_shank    , T_R_foot, T_R_toes ,
T_L_thigh, T_L_shank, T_L_foot, T_L_toes;

//Offset for dislocating CM
MoVector    offset_HAT,
            offset_R_thigh, offset_R_shank, offset_R_foot,
offset_R_toes,
            offset_L_thigh, offset_L_shank, offset_L_foot,
offset_L_toes;

offset_HAT = offset_R_thigh = offset_R_shank = offset_R_foot =
offset_R_toes = offset_L_thigh = offset_L_shank = offset_L_foot =
offset_L_toes = MoNullState;

//Mass elements
MoMassElement mass_HAT        (K25, m_HAT, T_HAT, offset_HAT)
            ;
MoMassElement mass_R_thigh    (K7, m_R_thigh, T_R_thigh,
offset_R_thigh)   ;
MoMassElement mass_R_shank    (K9, m_R_shank, T_R_shank,
offset_R_shank)   ;
MoMassElement mass_R_foot     (K11, m_R_foot, T_R_foot, offset_R_foot);
MoMassElement mass_R_toes     (K14, m_R_toes, offset_R_toes);
MoMassElement mass_L_thigh    (K16, m_L_thigh, T_L_thigh,
offset_L_thigh)   ;
MoMassElement mass_L_shank    (K18, m_L_shank, T_L_shank,
offset_L_shank)   ;
MoMassElement mass_L_foot     (K20, m_L_foot, T_L_foot, offset_L_foot)
            ;
MoMassElement mass_L_toes     (K23, m_L_toes, offset_L_toes);
```

```
//Creating the body
MoMapChain Body;
Body << xTrans << virt1
    << yTrans << virt2
    << Rot << virt3
    << HAT_rot << HAT
    << R_hip << L_hip
    << R_thigh << L_thigh
    << R_knee << L_knee
    << R_shank << L_shank
    << R_ankle << L_ankle
    << R_foot << L_foot
    << R_toes << L_toes
    << R_toe << L_toe
    << mass_HAT
    << mass_R_thigh << mass_L_thigh
    << mass_R_shank << mass_L_shank
    << mass_R_foot << mass_L_foot
    << mass_R_toes << mass_L_toes;

/*    (2)   MARKERS
/*********************************************************************
//All elements for watching the markers
MoFrame F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, F13, F14,
F15, F16, F17, F18, F19, F20, F21, F22, F23, F24, F25, F26, F27, F28,
F29, F30, F31, F32, F33, F34, F35, F36, F37, F38, F39, F40, F41, F42,
F43, F44, F45, F46, F47, F48;
MoLinearVariable peloTrX, peloTrY, rfeoTrX, rfeoTrY, rtioTrX, rtioTrY,
rtoeTrX, rtoeTrY, rheeTrX, rheeTrY, rtoetipTrX, rtoetipTrY, lfeoTrX,
lfeoTrY, ltioTrX, ltioTrY, ltoeTrX, ltoeTrY, lheeTrX, lheeTrY,
ltoetipTrX, ltoetipTrY, shouldersTrX, shouldersTrY;
MoVector l_peloX, l_peloY, l_rfeoX, l_rfeoY, l_rtioX, l_rtioY,
l_rtoeX, l_rtoeY, l_rheeX, l_rheeY, l_rtoetipX, l_rtoetipY, l_lfeoX,
l_lfeoY, l_ltioX, l_ltioY, l_ltoeX, l_ltoeY, l_lheeX, l_lheeY,
l_ltoetipX, l_ltoetipY, l_shouldersX, l_shouldersY;

MoElementaryJoint peloX      (K0, F1, peloTrX, xAxis);
MoElementaryJoint peloY      (F2, F3, peloTrY, yAxis);
MoElementaryJoint rfeoX      (K0, F5, rfeoTrX, xAxis);
MoElementaryJoint rfeoY      (F6, F7, rfeoTrY, yAxis);
MoElementaryJoint rtioX      (K0, F9, rtioTrX, xAxis);
MoElementaryJoint rtioY      (F10, F11, rtioTrY, yAxis);
MoElementaryJoint rtoeX      (K0, F13, rtoeTrX, xAxis);
MoElementaryJoint rtoeY      (F14, F15, rtoeTrY, yAxis);
MoElementaryJoint rheeX      (K0, F17, rheeTrX, xAxis);
MoElementaryJoint rheeY      (F18, F19, rheeTrY, yAxis);
MoElementaryJoint rtoetipX   (K0, F21, rtoetipTrX, xAxis);
MoElementaryJoint rtoetipY   (F22, F23, rtoetipTrY, yAxis);
MoElementaryJoint lfeoX      (K0, F25, lfeoTrX, xAxis);
MoElementaryJoint lfeoY      (F26, F27, lfeoTrY, yAxis);
```

```
MoElementaryJoint ltioX      (K0, F29, ltioTrX, xAxis);
MoElementaryJoint ltioY      (F30, F31, ltioTrY, yAxis);
MoElementaryJoint ltoeX      (K0, F33, ltoeTrX, xAxis);
MoElementaryJoint ltoeY      (F34, F35, ltoeTrY, yAxis);
MoElementaryJoint lheeX      (K0, F37, lheeTrX, xAxis);
MoElementaryJoint lheeY      (F38, F39, lheeTrY, yAxis);
MoElementaryJoint ltoetipX   (K0, F41, ltoetipTrX, xAxis);
MoElementaryJoint ltoetipY   (F42, F43, ltoetipTrY, yAxis);
MoElementaryJoint shouldersX (K0, F45, shouldersTrX, xAxis);
MoElementaryJoint shouldersY (F46, F47, shouldersTrY, yAxis);

MoRigidLink pelo1            (F1, F2, l_peloX);
MoRigidLink pelo2            (F3, F4, l_peloY);
MoRigidLink rfeo1            (F5, F6, l_rfeoX);
MoRigidLink rfeo2            (F7, F8, l_rfeoY);
MoRigidLink rtio1            (F9, F10, l_rtioX);
MoRigidLink rtio2            (F11, F12, l_rtioY);
MoRigidLink rtoe1            (F13, F14, l_rtoeX);
MoRigidLink rtoe2            (F15, F16, l_rtoeY);
MoRigidLink rhee1            (F17, F18, l_rheeX);
MoRigidLink rhee2            (F19, F20, l_rheeY);
MoRigidLink rtoetip1         (F21, F22, l_rtoetipX);
MoRigidLink rtoetip2         (F23, F24, l_rtoetipY);
MoRigidLink lfeo1            (F25, F26, l_lfeoX);
MoRigidLink lfeo2            (F27, F28, l_lfeoY);
MoRigidLink ltio1            (F29, F30, l_ltioX);
MoRigidLink ltio2            (F31, F32, l_ltioY);
MoRigidLink ltoe1            (F33, F34, l_ltoeX);
MoRigidLink ltoe2            (F35, F36, l_ltoeY);
MoRigidLink lhee1            (F37, F38, l_lheeX);
MoRigidLink lhee2            (F39, F40, l_lheeY);
MoRigidLink ltoetip1         (F41, F42, l_ltoetipX);
MoRigidLink ltoetip2         (F43, F44, l_ltoetipY);
MoRigidLink shoulders1       (F45, F46, l_shouldersX);
MoRigidLink shoulders2       (F47, F48, l_shouldersY);

MoMapChain Markers;
Markers << peloX << shouldersX
        << rfeoX << rtioX << rtoeX << rheeX << rtoetipX
        << lfeoX << ltioX << ltoeX << lheeX << ltoetipX
        << pelo1 << shoulders1
        << rfeo1 << rtio1 << rtoe1 << rhee1 << rtoetip1
        << lfeo1 << ltio1 << ltoe1 << lhee1 << ltoetip1
        << peloY << shouldersY
        << rfeoY << rtioY << rtoeY << rheeY << rtoetipY
        << lfeoY << ltioY << ltoeY << lheeY << ltoetipY
        << pelo2 << shoulders2
        << rfeo2 << rtio2 << rtoe2 << rhee2 << rtoetip2
        << lfeo2 << ltio2 << ltoe2 << lhee2 << ltoetip2    ;
```

```
l_peloX = l_peloY = l_shouldersX = l_shouldersY =
l_rfeoX = l_rfeoY = l_rtioX = l_rtioY = l_rtoeX = l_rtoeY = l_rheeX =
l_rheeY = l_rtoetipX = l_rtoetipY =
l_lfeoX = l_lfeoY = l_ltioX = l_ltioY = l_ltoeX = l_ltoeY = l_lheeX =
l_lheeY = l_ltoetipX = l_ltoetipY = MoNullState;

/*    (3)   CAMERAS
/*****************************************************************
MoFrame C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C13, C14;
MoLinearVariable HIP, RKNEE, RANKLE, RTOE, LKNEE, LANKLE, LTOE;
MoVector l_HIP, l_RKNEE, l_RANKLE, l_RTOE, l_LKNEE, l_LANKLE, l_LTOE;

l_HIP = l_RKNEE = l_RANKLE = l_RTOE = l_LKNEE = l_LANKLE = l_LTOE =
MoNullState;

MoElementaryJoint prism1      (K0,  C1,         HIP,        xAxis);
MoElementaryJoint prism2      (K0,  C3,         RKNEE,
      xAxis);
MoElementaryJoint prism3      (K0,  C5,         RANKLE,
      xAxis);
MoElementaryJoint prism4      (K0,  C7,         RTOE,       xAxis);
MoElementaryJoint prism5      (K0,  C9,         LKNEE,
      xAxis);
MoElementaryJoint prism6      (K0,  C11,  LANKLE,           xAxis);
MoElementaryJoint prism7      (K0,  C13,  LTOE,       xAxis);

MoRigidLink camera1                (C1, C2, l_HIP);
MoRigidLink camera2                (C3, C4, l_RKNEE);
MoRigidLink camera3                (C5, C6, l_RANKLE);
MoRigidLink camera4                (C7, C8, l_RTOE);
MoRigidLink camera5                (C9, C10, l_LKNEE);
MoRigidLink camera6                (C11, C12, l_LANKLE);
MoRigidLink camera7                (C13, C14, l_LTOE);

MoMapChain Cameras;
Cameras << prism1 << prism2 << prism3 << prism4 << prism5 << prism6 <<
prism7
        << camera1 << camera2 << camera3 << camera4 << camera5 <<
camera6 << camera7;

/*****************************************************************
*                      DIMENSIONS & INITIAL CONDITIONS            *
/*****************************************************************
/*    (1) Lengths
/*****************************************************************
l_virt1 = l_virt2 = l_virt3 = l_HAT =  l_R_thigh = l_R_shank =
l_R_toes = l_L_thigh = l_L_shank = l_L_toes = MoNullState;
MoReal l_R_foot_x1, l_R_foot_x2, l_R_foot_y, l_L_foot_x1, l_L_foot_x2,
l_L_foot_y;
```

```
ifstream input_length ( 'Data/dimensions&masses/lengths.dat' ) ;
input_length >> l_HAT.y
    >> l_R_thigh.y >> l_R_shank.y >> l_R_foot_y >> l_R_foot_x1 >>
l_R_foot_x2 >> l_R_toes.x
    >> l_L_thigh.y >> l_L_shank.y >> l_L_foot_y >> l_L_foot_x1 >>
l_L_foot_x2 >> l_L_toes.x;
input_length.close () ;
l_R_foot[0] = MoVector ( l_R_foot_x1 , l_R_foot_y , 0 );
l_R_foot[1] = MoVector ( l_R_foot_x2 , l_R_foot_y , 0 );
l_L_foot[0] = MoVector ( l_L_foot_x1 , l_L_foot_y , 0 );
l_L_foot[1] = MoVector ( l_L_foot_x2 , l_L_foot_y , 0 );

/*    (2) Masses of all links
/*****************************************************************
ifstream input_masses ( 'Data/dimensions&masses/masses.dat' ) ;
input_masses >> m_HAT
    >> m_R_thigh >> m_R_shank >> m_R_foot >> m_R_toes
    >> m_L_thigh >> m_L_shank >> m_L_foot >> m_L_toes;
input_masses.close () ;

/*    (3) Offsets for dislocating CM
/*****************************************************************
MoReal      off_R_HAT,
        off_R_thigh, off_R_shank, off_R_foot_x, off_R_foot_y,
off_R_toes,
        off_L_thigh, off_L_shank, off_L_foot_x, off_L_foot_y,
off_L_toes;

ifstream input_offset ( 'Data/dimensions&masses/offset.dat' ) ;
input_offset >> off_R_HAT
    >>      off_R_thigh >> off_R_shank >> off_R_foot_x >> off_R_foot_y
>> off_R_toes
    >>      off_L_thigh >> off_L_shank >> off_L_foot_x >> off_L_foot_y
>> off_L_toes;
input_offset.close () ;

offset_HAT        = l_HAT           *     off_R_HAT;
offset_R_thigh    = l_R_thigh       *     off_R_thigh;
offset_R_shank    = l_R_shank       *     off_R_shank;
offset_R_foot     = MoVector (l_R_foot_x2 * off_R_foot_x,   l_R_foot_y
* off_R_foot_y,   0);
offset_R_toes     = l_R_toes        *     off_R_toes;
offset_L_thigh    = l_L_thigh       *     off_L_thigh;
offset_L_shank    = l_L_shank       *     off_L_shank;
offset_L_foot     = MoVector (l_L_foot_x2 * off_L_foot_x,   l_L_foot_y
* off_L_foot_y,   0);
offset_L_toes     = l_L_toes        *     off_L_toes;

/*    (4) Moment of inertia
/*****************************************************************
```

```
MoReal tensor_hat, tensor_R_thigh, tensor_R_shank, tensor_R_foot,
tensor_R_toes, tensor_L_thigh, tensor_L_shank, tensor_L_foot,
tensor_L_toes;

ifstream input_inertia ( 'Data/dimensions&masses/inertia.dat' ) ;
input_inertia >> tensor_hat
    >> tensor_R_thigh >> tensor_R_shank >> tensor_R_foot >>
tensor_R_toes
    >> tensor_L_thigh >> tensor_L_shank >> tensor_L_foot >>
tensor_L_toes;
input_inertia.close();

T_HAT             =       MoInertiaTensor( 0 , 0 , tensor_hat)      ;
T_R_thigh         =       MoInertiaTensor( 0 , 0 , tensor_R_thigh ) ;
T_R_shank         =       MoInertiaTensor( 0 , 0 , tensor_R_shank ) ;
T_R_foot          =       MoInertiaTensor( 0 , 0 , tensor_R_foot )  ;
T_R_toes          =       MoInertiaTensor( 0 , 0 , tensor_R_toes )  ;
T_L_thigh         =       MoInertiaTensor( 0 , 0 , tensor_L_thigh ) ;
T_L_shank         =       MoInertiaTensor( 0 , 0 , tensor_L_shank ) ;
T_L_foot          =       MoInertiaTensor( 0 , 0 , tensor_L_foot )  ;
T_L_toes          =       MoInertiaTensor( 0 , 0 , tensor_L_toes )  ;

/*********************************************************************
*                 READING THE INITIAL DATA FROM INPUT FILE          *
*********************************************************************
//Initial angles and positions
// (1) BODY
ifstream input_init ( 'Data/body_init.dat' ) ;
input_init
      >> trX.q >> trY.q >> rotZ.q
      >> ang1.q >> ang2.q
      >> ang3.q >> ang4.q
      >> ang5.q >> ang6.q
      >> ang7.q >> ang8.q
      >> angHAT.q;
input_init.close () ;

// (2) MARKERS
ifstream input_markers ( 'Data/markers_init.dat' ) ;
input_markers
      >> peloTrX.q >> peloTrY.q
      >> rfeoTrX.q >> rfeoTrY.q
      >> rtioTrX.q >> rtioTrY.q
      >> rtoeTrX.q >> rtoeTrY.q
      >> rheeTrX.q >> rheeTrY.q
      >> rtoetipTrX.q >> rtoetipTrY.q
      >> lfeoTrX.q >> lfeoTrY.q
      >> ltioTrX.q >> ltioTrY.q
      >> ltoeTrX.q >> ltoeTrY.q
```

```
      >> lheeTrX.q >> lheeTrY.q
      >> ltoetipTrX.q >> ltoetipTrY.q
      >> shouldersTrX.q >> shouldersTrY.q ;
input_markers.close () ;

// (3) CAMERAS
ifstream input_cameras ( 'Data/cameras_init.dat' ) ;
input_cameras
    >> HIP.q >> RKNEE.q >> RANKLE.q >> RTOE.q >> LKNEE.q >> LANKLE.q
>> LTOE.q   ;
input_cameras.close () ;

// (4) ANIMATION
Body.doMotion(DO_POSITION)      ;
Markers.doMotion(DO_POSITION) ;
Cameras.doMotion(DO_POSITION) ;

/*********************************************************************
*                 READING THE INPUT FILE for the variables          *
*********************************************************************
//Variables inserted as input
MoVariableList varInputs;
varInputs << trX << trY << rotZ
    << ang1 << ang2 << ang3 << ang4 << ang5 << ang6 << ang7 << ang8 <<
angHAT;

//name of the input file
string inFileName('Data/body.dat');

// constructor different than MoVarListInput
MoInterpolatingVarListInput varListInput(varInputs, OUTPUT_POSITION |
OUTPUT_VELOCITY | OUTPUT_ACCELERATION, inFileName);
      //interpolating because the file gives discontinuous points

MoMapChain inputChain;
inputChain << varListInput;

/*********************************************************************
*                 READING THE INPUT FILE for the markers            *
*********************************************************************
//Creating the list of inputs
MoVariableList varInputsMarkers;
varInputsMarkers << peloTrX << peloTrY
                 << rfeoTrX << rfeoTrY << rtioTrX << rtioTrY
                 << rtoeTrX << rtoeTrY << rheeTrX << rheeTrY <<
rtoetipTrX << rtoetipTrY
                 << lfeoTrX << lfeoTrY << ltioTrX << ltioTrY
                 << ltoeTrX << ltoeTrY << lheeTrX << lheeTrY <<
ltoetipTrX << ltoetipTrY
                 << shouldersTrX << shouldersTrY ;
```

```
//Name of the input file
string inMarkers('Data/markers.dat');                //name of the
input file

//Constructor different than MoVarListInput

//ONLY THE POSITIONS OF THE MARKERS
MoInterpolatingVarListInput varListInputMarkers(varInputsMarkers,
OUTPUT_POSITION, inMarkers);
//interpolating because the file gives discontinuous points

MoMapChain inputMarkers;
inputMarkers << varListInputMarkers;

/*********************************************************************
*                 READING THE INPUT FILE  for the cameras            *
*********************************************************************/

//Variables inserted as input
MoVariableList varInputsCameras;
varInputsCameras << HIP << RKNEE << RANKLE << RTOE << LKNEE << LANKLE
<< LTOE;

//name of the input file
string inCameras('Data/cameras.dat');

//constructor different than MoVarListInput
MoInterpolatingVarListInput varListInputsCameras(varInputsCameras,
OUTPUT_POSITION, inCameras);
//interpolating because the file gives discontinuous points

MoMapChain inputCameras;
inputCameras << varListInputsCameras;

/*********************************************************************
*                        INTEGRATOR & VISUALIZATION                  *
/*********************************************************************
/*    Visualizing the floor as a grid
/*********************************************************************
MoFrame Kplane;
MoRotationMatrix dR = MoXRotationMatrix(-90.0 * DEG_TO_RAD);
MoRigidLink rot_plane (K0, Kplane, dR);

Body << rot_plane;
/*********************************************************************

MoReal dT = 0.02 ;

MyClock clock(dT);
```

```
MoMapChain animationChain;                          //new MoMapChain object
AnimationChain    << inputChain << Body
//add the integrated model and the output list to the new MoMapChain
object
                  << inputMarkers << Markers
                  << inputCameras << Cameras
                  << clock;

MoScene Scene ( animationChain ) ;        // interface for 3D-rendering

//Scale factors
MoReal scale = 0.2, frameScale = 0.1, massScale = 0.08, markerScale =
0.1;

Scene.setSelectorMotion(DO_ALL);

//View links & joints
Scene.makeShape (R_hip, scale, scale, scale) ;
Scene.makeShape (R_knee, scale, scale, scale) ;
Scene.makeShape (R_ankle, scale, scale, scale) ;
Scene.makeShape (R_toes, frameScale, scale, frameScale) ;
Scene.makeShape (L_hip, scale, scale, scale) ;
Scene.makeShape (L_knee, scale, scale, scale) ;
Scene.makeShape (L_ankle, scale, scale, scale) ;
Scene.makeShape (HAT_rot, scale, scale, scale) ;
Scene.makeShape (L_toes, frameScale, scale, frameScale) ;
Scene.makeShape (R_hip, R_thigh, scale, scale, scale) ;
Scene.makeShape (R_knee, R_shank, scale, scale, scale) ;
Scene.makeShape (L_hip, L_thigh, scale, scale, scale) ;
Scene.makeShape (L_knee, L_shank, scale, scale, scale) ;
Scene.makeShape (HAT_rot, HAT, scale, scale, scale) ;

//View geometries (=feet+floor+markers)
Scene.makeShape(K11, '/Geometry/footGeometryGreen.so', frameScale,
frameScale, frameScale);            //Green for right
Scene.makeShape(K14, '/Geometry/toesGeometryGreen.so', frameScale,
frameScale, frameScale);
Scene.makeShape(K20, '/Geometry/footGeometryRed.so', frameScale,
frameScale, frameScale);            //Red for left
Scene.makeShape(K23, '/Geometry/toesGeometryRed.so', frameScale,
frameScale, frameScale);
Scene.makeShape(K0,  '/Geometry/floor.so');                 //floor
Scene.makeShape(F4,  '/Geometry/markerBlue.so', markerScale,
markerScale, markerScale);          //PELO
Scene.makeShape(F8,  '/Geometry/markerGreen.so', markerScale,
markerScale, markerScale);          //RFEO
Scene.makeShape(F12, '/Geometry/markerGreen.so', markerScale,
markerScale, markerScale);          //RTIO
Scene.makeShape(F16, '/Geometry/markerGreen.so', markerScale,
markerScale, markerScale);          //RTOE
```

```
Scene.makeShape(F20, '/Geometry/markerGreen.so', markerScale,
markerScale, markerScale);          //RHEE
Scene.makeShape(F24, '/Geometry/markerGreen.so', markerScale,
markerScale, markerScale);          //RTOETIP
Scene.makeShape(F28, '/Geometry/markerRed.so', markerScale,
markerScale, markerScale);          //LFEO
Scene.makeShape(F32, '/Geometry/markerRed.so', markerScale,
markerScale, markerScale);          //LTIO
Scene.makeShape(F36, '/Geometry/markerRed.so', markerScale,
markerScale, markerScale);          //LTOE
Scene.makeShape(F40, '/Geometry/markerRed.so', markerScale,
markerScale, markerScale);          //LHEE
Scene.makeShape(F44, '/Geometry/markerRed.so', markerScale,
markerScale, markerScale);          //LTOETIP
Scene.makeShape(F48, '/Geometry/markerBlue.so', markerScale,
markerScale, markerScale);          //SHOULDERS

//view frames
Scene.makeShape(K0,  'MoFrameGeom.so', frameScale, frameScale,
frameScale);
//show the origin

//Cameras
Scene.makeCamera(C2, 'Hip');              //camera fixe to the frame C2
Scene.makeCamera(C4, 'Right knee');       //camera fixe to the frame C4
Scene.makeCamera(C6, 'Right ankle');      //camera fixe to the frame C6
Scene.makeCamera(C8, 'Right toe');        //camera fixe to the frame C8
Scene.makeCamera(C10, 'Left knee');       //camera fixe to the frame
C10
Scene.makeCamera(C12, 'Left ankle');      //camera fixe to the frame
C12
Scene.makeCamera(C14, 'Left toe');        //camera fixe to the frame
C14

//view masses
Scene.makeShape(mass_HAT,      'MoMassElement.so', massScale);
Scene.makeShape(mass_R_thigh,  'MoMassElement.so', massScale);
Scene.makeShape(mass_R_shank,  'MoMassElement.so', massScale);
Scene.makeShape(mass_R_foot,   'MoMassElement.so', massScale);
Scene.makeShape(mass_R_toes,   'MoMassElement.so', massScale*0.8);
Scene.makeShape(mass_L_thigh,  'MoMassElement.so', massScale);
Scene.makeShape(mass_L_shank,  'MoMassElement.so', massScale);
Scene.makeShape(mass_L_shank,  'MoMassElement.so', massScale);
Scene.makeShape(mass_L_foot,   'MoMassElement.so', massScale);
Scene.makeShape(mass_L_toes,   'MoMassElement.so', massScale*0.8);

Scene.addAnimationObject ( animationChain ) ;   // animate
animationChain
Scene.setAnimationIncrement ( dT ) ;            // animate as fast as
possible

MoWidget widget(Scene, animationChain, 'widget');
```

```
widget.addSeparator();
widget.addLabel(ang1.q, 'ang1.q');
widget.addLabel(ang1.qd, 'ang1.qd');
widget.addLabel(ang1.qdd, 'ang1.qdd');
widget.addSeparator();
widget.addLabel(animationChain.integratorTime->modelTime, 'Time: ');

Scene.show() ;
MoScene::mainLoop() ;                              // move the scene

return 0 ;
}
}
```

APPENDIX III

ANTHROPOMETRIC SCRIPT FOR MATLAB

```
% ANTHROPOMETRIC DATA
%
% Author:   Philippe Ferreira
% Date:     08.12.2011
%
%-------------------------------------------------------------------
%
% This script is intended to be used with the MBS model 'HumanBody',
% implemented on MOBILE.
% It creates the input data for:
%       1) the segment lengths,
%       2) the masses,
%       3) the position of the CM, and
%       4) the moments of inertia.
%
%-------------------------------------------------------------------

% !Tmass -> a Tmass must be defined as a global variable,
corresponding to
% the total mass of the subject.

loadc3d                         % choose the input c3d file
s = size(ans.PELO,1);           % scalar s = number of samples

%---%---------------------------------------------------------------%
% 1 % LENGTHS                                                      %
%---%---------------------------------------------------------------%

% Creating the vector with the position (x, y) that is the middle
point of
% the segment formed by the shoulder articulations (RHUP and LHUP).

glenohumeral_joint = zeros (s, 2);

for i=1:1:s,
% Creating a point in the middle of the shoulders
    glenohumeral_joint(i,1) = ( ans.RHUP(i,2) + ans.LHUP(i,2) ) / 2 ;
    glenohumeral_joint(i,2) = ( ans.RHUP(i,3) + ans.LHUP(i,3) ) / 2 ;
end

lengths = zeros(13,1);

%mean = sum of the parts / number of parts

for i=1:1:s,
% Sum
    lengths(1, 1) = lengths(1, 1) - sqrt( (glenohumeral_joint(i,1) -
ans.PELO(i,2))^2 + (glenohumeral_joint(i,2) - ans.PELO(i,3))^2 )/1000;
```

```
    lengths(2, 1) = lengths(2, 1) - sqrt( (ans.PELO(i,2) -
ans.RFEO(i,2))^2 + (ans.PELO(i,3) - ans.RFEO(i,3))^2 )/1000;
    lengths(3, 1) = lengths(3, 1) - sqrt( (ans.RFEO(i,2) -
ans.RTIO(i,2))^2 + (ans.RFEO(i,3) - ans.RTIO(i,3))^2 )/1000;
    lengths(4, 1) = lengths(4, 1) + ( -0.0430229 );
    lengths(5, 1) = lengths(5, 1) + (  0.048724 );
    lengths(6, 1) = lengths(6, 1) + ( -0.134276 );
    lengths(7, 1) = lengths(7, 1) - sqrt( (ans.RTOETIP(i,2) -
ans.RTOE(i,2))^2 + (ans.RTOETIP(i,3) - ans.RTOE(i,3))^2 )/1000;
    lengths(8, 1) = lengths(8, 1) - sqrt( (ans.PELO(i,2) -
ans.LFEO(i,2))^2 + (ans.PELO(i,3) - ans.LFEO(i,3))^2 )/1000;
    lengths(9, 1) = lengths(9, 1) - sqrt( (ans.LFEO(i,2) -
ans.LTIO(i,2))^2 + (ans.LFEO(i,3) - ans.LTIO(i,3))^2 )/1000;
    lengths(10,1) = lengths(10,1) + ( -0.0430229 );
    lengths(11,1) = lengths(11,1) + (  0.048724 );
    lengths(12,1) = lengths(12,1) + ( -0.134276 );
    lengths(13,1) = lengths(13,1) - sqrt( (ans.LTOETIP(i,2) -
ans.LTOE(i,2))^2 + (ans.LTOETIP(i,3) - ans.LTOE(i,3))^2 )/1000;
end
lengths = lengths / s ;
% Division


save
('C:\MobileHome\examples\04_HumanBody\Data\dimensions&masses\lengths.d
at', 'lengths', '-ASCII');


%---%-------------------------------------------------------------------%
% 2 % MASSES                                                          %
%---%-------------------------------------------------------------------%


% Winter DA. Biomechanics and Motor Control of Human Movement, pp. 86,
4th Ed., Wiley, 2009.


masses = zeros(9,1);


masses(1, 1) = Tmass * 0.678  ;     % HAT
masses(2, 1) = Tmass * 0.100  ;     % R_thigh
masses(3, 1) = Tmass * 0.0465 ;     % R_shank
masses(4, 1) = Tmass * 0.0130 ;     % R_foot !originally 0.0145
masses(5, 1) = Tmass * 0.0015 ;     % R_toes !does not exist in the
literature
masses(6, 1) = Tmass * 0.100  ;     % L_thigh
masses(7, 1) = Tmass * 0.0465 ;     % L_shank
masses(8, 1) = Tmass * 0.0130 ;     % L_foot !originally 0.0145
masses(9, 1) = Tmass * 0.0015 ;     % L_toes !does not exist in the
literature


save
('C:\MobileHome\examples\04_HumanBody\Data\dimensions&masses\masses.da
t', 'masses', '-ASCII');
```

```
%---%-------------------------------------------------------------------%
% 3 % OFFSET'S FOR THE CENTERS OF MASS                                  %
%---%-------------------------------------------------------------------%

% Winter DA. Biomechanics and Motor Control of Human Movement, pp. 86,
4th Ed., Wiley, 2009.

% this file has only the coefficients taken from Winter's book

offset = zeros(11,1);

offset( 1, 1) = 0.626 ;             % HAT
offset( 2, 1) = 0.433 ;             % R_thigh
offset( 3, 1) = 0.433 ;             % R_shank
offset( 4, 1) = 0.5   ;             % R_foot_x
offset( 5, 1) = 0.5   ;             % R_foot_y
offset( 6, 1) = 0.5   ;             % R_toes >> assumed
offset( 7, 1) = 0.433 ;             % L_thigh
offset( 8, 1) = 0.433 ;             % L_shank
offset( 9, 1) = 0.5   ;             % L_foot_x
offset(10, 1) = 0.5   ;             % L_foot_y
offset(11, 1) = 0.5   ;             % L_toes >> assumed

save
('C:\MobileHome\examples\04_HumanBody\Data\dimensions&masses\offset.da
t', 'offset', '-ASCII');

%---%-------------------------------------------------------------------%
% 4 % MOMENTS OF INERTIA                                                %
%---%-------------------------------------------------------------------%

% Winter DA. Biomechanics and Motor Control of Human Movement, pp. 86,
4th Ed., Wiley, 2009.

inertia = zeros(9,1);

inertia( 1, 1) = masses( 1, 1) * ( 0.496 * lengths (1,1)  )^2 ; % HAT
inertia( 2, 1) = masses( 2, 1) * ( 0.323 * lengths (2,1)  )^2 ; % R_thigh
inertia( 3, 1) = masses( 3, 1) * ( 0.302 * lengths (3,1)  )^2 ; % R_shank
inertia( 4, 1) = masses( 4, 1) * ( 0.475 * 0.141001       )^2 ; % R_foot
inertia( 5, 1) = masses( 5, 1) * ( 0.000 * lengths (7,1)  )^2 ; % R_toe
inertia( 6, 1) = masses( 6, 1) * ( 0.323 * lengths (8,1)  )^2 ; % L_thigh
inertia( 7, 1) = masses( 7, 1) * ( 0.302 * lengths (9,1)  )^2 ; % L_shank
inertia( 8, 1) = masses( 8, 1) * ( 0.475 * 0.141001       )^2 ; % L_foot
inertia( 9, 1) = masses( 9, 1) * ( 0.000 * lengths (13,1) )^2 ; % L_toe
```

```
save
('C:\MobileHome\examples\04_HumanBody\Data\dimensions&masses\inertia.d
at', 'inertia', '-ASCII');

%---%-------------------------------------------------------------------%
% 5 % RADIUS OF THE SPHERES                                           %
%---%-------------------------------------------------------------------%

spheres = zeros(6, 1);

spheres( 1, 1) = min( ans.RHEE(1:s,3)    )  /1000 ;
spheres( 2, 1) = min( ans.RTOE(1:s,3)    )  /1000 ;
spheres( 3, 1) = min( ans.RTOETIP(1:s,3) )  /1000 ;
spheres( 4, 1) = min( ans.LHEE(1:s,3)    )  /1000 ;
spheres( 5, 1) = min( ans.LTOE(1:s,3)    )  /1000 ;
spheres( 6, 1) = min( ans.LTOETIP(1:s,3) )  /1000 ;

save
('C:\MobileHome\examples\04_HumanBody\Data\contact_model\spheres.dat',
'spheres', '-ASCII');
```

APPENDIX IV

FOOT GEOMETRY CODE

```
#Inventor V2.0 ascii

Separator {
      Separator {
            Normal {
                  vector [-0.661890918    -0.7496001685      0.0,
                          -0.3051268422    0.952311719       0.0,
                           0.0            -1.0               0.0,
                           0.0             0.0               1.0,
                           0.0             0.0              -1.0]
            }
            NormalBinding {
                  value PER_FACE
            }
            Material {
                  diffuseColor 0 1 0      # r g b
                  transparency      0 ~
            }
            Coordinate3 {
                  point [ 0.0       0.0          0.25,
                          0.0       0.0         -0.25,
                          0.48724  -0.430229    -0.25,
                          0.48724  -0.430229     0.25,
                          0.0       0.0          0.25,
                          0.0       0.0         -0.25,
                         -1.34276  -0.430229    -0.25,
                         -1.34276  -0.430229     0.25,
                         -1.34276  -0.430229     0.25,
                          0.48724  -0.430229     0.25,
                          0.48724  -0.430229    -0.25,
                         -1.34276  -0.430229    -0.25,
                          0.0       0.0          0.25,
                         -1.34276  -0.430229     0.25,
                          0.48724  -0.430229     0.25,
                          0.0       0.0         -0.25,
                         -1.34276  -0.430229    -0.25,
                          0.48724  -0.430229    -0.25,
                        ]
            }
            FaceSet {
                  numVertices [ 4, 4, 4, 3, 3]
            }
   }

}
```

APPENDIX V

ANGLE CALCULATION IN MATLAB

APPENDIX V.A – READC3D SCRIPT

This script creates the input files used by the *Reader*.

```
loadc3d                          %choose the input c3d file
s = size(ans.PELO,1);            %scalar s = number of samples
dT = 0.01;
A = zeros(s,37);                 %final matrix

% Creating the vector with the position (x, y) that is the middle
point of
% the segment formed by the shoulder articulations (RHUP and LHUP).

glenohumeral_joint = zeros (s, 2);

for i=1:1:s,
% Creating a point in the middle of the shoulders
    glenohumeral_joint(i,1) = ( ans.RHUP(i,2) + ans.LHUP(i,2) ) / 2 ;
    glenohumeral_joint(i,2) = ( ans.RHUP(i,3) + ans.LHUP(i,3) ) / 2 ;
end

for i=1:1:s,                     %insert the data from the c3d file
    A(i,1) = dT*i-dT;   %time vector
    A(i,2) = ans.PELO(i,2)/1000;    %translation in the Y-direction
    A(i,5) = ans.PELO(i,3)/1000;    %translation in the Z-direction
    A(i,8) = 0.0;    %rotation of the all model is always null

    %calculating the angles using the plots from the c3d file
    A(i,11) = -atan((ans.RFEO(i,2)-ans.PELO(i,2))/(ans.RFEO(i,3)-
ans.PELO(i,3)));
     % R_hip
    A(i,14) = -(atan((ans.RTIO(i,2)-ans.RFEO(i,2))/(ans.RTIO(i,3)-
     ans.RFEO(i,3)))+A(i,11)); % R_knee
    A(i,17) = atan((ans.RHEE(i,3)-ans.RTOE(i,3))/(ans.RHEE(i,2)-
ans.RTOE(i,2)))-
     A(i,14)-A(i,11); % R_ankle
    A(i,20) = atan((ans.RTOETIP(i,3)-ans.RTOE(i,3))/(ans.RTOETIP(i,2)-
     ans.RTOE(i,2)))-A(i,17)-A(i,14)-A(i,11); % R_toes
    A(i,23) = -atan((ans.LFEO(i,2)-ans.PELO(i,2))/(ans.LFEO(i,3)-
ans.PELO(i,3)));
     % L_hip
    A(i,26) = -(atan((ans.LTIO(i,2)-ans.LFEO(i,2))/(ans.LTIO(i,3)-
     ans.LFEO(i,3)))+A(i,23)); % L_knee
    A(i,29) = (atan((ans.LHEE(i,3)-ans.LTOE(i,3))/(ans.LHEE(i,2)-
     ans.LTOE(i,2))))-A(i,26)-A(i,23); % L_ankle
    A(i,32) = atan((ans.LTOETIP(i,3)-ans.LTOE(i,3))/(ans.LTOETIP(i,2)-
     ans.LTOE(i,2)))-A(i,29)-A(i,26)-A(i,23); %L_toes
```

```
    A(i,35) = pi - atan( (glenohumeral_joint(i,1) - (ans.PELO(i,2))) /
(
     glenohumeral_joint(i,2)- ans.PELO(i,3)) ); %HAT
end

for i=2:1:s,                    %differentiate the positions to obtain
the velocities
    A(i,3)  = (A(i,2)-A(i-1,2))/dT;
    A(i,6)  = (A(i,5)-A(i-1,5))/dT;
    A(i,9)  = (A(i,8)-A(i-1,8))/dT;
    A(i,12) = (A(i,11)-A(i-1,11))/dT;
    A(i,15) = (A(i,14)-A(i-1,14))/dT;
    A(i,18) = (A(i,17)-A(i-1,17))/dT;
    A(i,21) = (A(i,20)-A(i-1,20))/dT;
    A(i,24) = (A(i,23)-A(i-1,23))/dT;
    A(i,27) = (A(i,26)-A(i-1,26))/dT;
    A(i,30) = (A(i,29)-A(i-1,29))/dT;
    A(i,33) = (A(i,32)-A(i-1,32))/dT;
    A(i,36) = (A(i,35)-A(i-1,35))/dT;
end

for i=3:1:s,                    %differentiate the velocities to obtain
the accelerations
    A(i,4)  = (A(i,3)-A(i-1,3))/dT;
    A(i,7)  = (A(i,6)-A(i-1,6))/dT;
    A(i,10) = (A(i,9)-A(i-1,9))/dT;
    A(i,13) = (A(i,12)-A(i-1,12))/dT;
    A(i,16) = (A(i,15)-A(i-1,15))/dT;
    A(i,19) = (A(i,18)-A(i-1,18))/dT;
    A(i,22) = (A(i,21)-A(i-1,21))/dT;
    A(i,25) = (A(i,24)-A(i-1,24))/dT;
    A(i,28) = (A(i,27)-A(i-1,27))/dT;
    A(i,31) = (A(i,30)-A(i-1,30))/dT;
    A(i,34) = (A(i,33)-A(i-1,33))/dT;
    A(i,37) = (A(i,36)-A(i-1,36))/dT;
end

save ('C:\MobileHome\examples\04_HumanBody\Data\body.dat', 'A', '-
ASCII');

B = zeros(1,12);

B(1,1) = A(1,2);
B(1,2) = A(1,5);
B(1,3) = A(1,8);
B(1,4) = A(1,11);
B(1,5) = A(1,14);
B(1,6) = A(1,17);
B(1,7) = A(1,20);
```

```
B(1,8) = A(1,23);
B(1,9) = A(1,26);
B(1,10) = A(1,29);
B(1,11) = A(1,32);
B(1,12) = A(1,35);
save ('C:\MobileHome\examples\04_HumanBody\Data\body_init.dat', 'B',
'-ASCII');

%-----------------------------------------------------------------------

markers = zeros(s,25);              %final matrix

for i=1:1:s,                        %insert the data from the c3d file
    markers(i,1)  = dT*i-dT;
    markers(i,2)  = ans.PELO(i,2)/1000;
    markers(i,3)  = ans.PELO(i,3)/1000;
    markers(i,4)  = ans.RFEO(i,2)/1000;
    markers(i,5)  = ans.RFEO(i,3)/1000;
    markers(i,6)  = ans.RTIO(i,2)/1000;
    markers(i,7)  = ans.RTIO(i,3)/1000;
    markers(i,8)  = ans.RTOE(i,2)/1000;
    markers(i,9)  = ans.RTOE(i,3)/1000;
    markers(i,10) = ans.RHEE(i,2)/1000;
    markers(i,11) = ans.RHEE(i,3)/1000;
    markers(i,12) = ans.RTOETIP(i,2)/1000;
    markers(i,13) = ans.RTOETIP(i,3)/1000;
    markers(i,14) = ans.LFEO(i,2)/1000;
    markers(i,15) = ans.LFEO(i,3)/1000;
    markers(i,16) = ans.LTIO(i,2)/1000;
    markers(i,17) = ans.LTIO(i,3)/1000;
    markers(i,18) = ans.LTOE(i,2)/1000;
    markers(i,19) = ans.LTOE(i,3)/1000;
    markers(i,20) = ans.LHEE(i,2)/1000;
    markers(i,21) = ans.LHEE(i,3)/1000;
    markers(i,22) = ans.LTOETIP(i,2)/1000;
    markers(i,23) = ans.LTOETIP(i,3)/1000;
    markers(i,24) = glenohumeral_joint(i,1)/1000;
    markers(i,25) = glenohumeral_joint(i,2)/1000;
end

save ('C:\MobileHome\examples\04_HumanBody\Data\markers.dat',
'markers', '-ASCII');

markers0 = zeros(1,24);

for i=1:1:24,
    markers0(1,i) = markers(1,i+1);
end
```

```
save ('C:\MobileHome\examples\04_HumanBody\Data\markers_init.dat',
'markers0', '-ASCII');

%-----------------------------------------------------------------------
cameras = zeros(s,8);

for i=1:1:s,
    cameras(i,1) = dT*i-dT;
    cameras(i,2) = ans.PELO(i,2)/1000;
    cameras(i,3) = ans.RFEO(i,2)/1000;
    cameras(i,4) = ans.RTIO(i,2)/1000;
    cameras(i,5) = ans.RTOE(i,2)/1000;
    cameras(i,6) = ans.LFEO(i,2)/1000;
    cameras(i,7) = ans.LTIO(i,2)/1000;
    cameras(i,8) = ans.LTOE(i,2)/1000;
end

save ('C:\MobileHome\examples\04_HumanBody\Data\cameras.dat',
'cameras', '-ASCII');

cameras0 = zeros(1,7);

    cameras0(1,1) = cameras(1,1);
    cameras0(1,2) = cameras(1,2);
    cameras0(1,3) = cameras(1,3);
    cameras0(1,4) = cameras(1,4);
    cameras0(1,5) = cameras(1,5);
    cameras0(1,6) = cameras(1,6);
    cameras0(1,7) = cameras(1,7);

save ('C:\MobileHome\examples\04_HumanBody\Data\cameras_init.dat',
'cameras0', '-ASCII');

%-----------------------------------------------------------------------
moments = zeros(s,1);

for i=1:1:s,
   moments(i,1) = ans.RAnkleMoment(i,1)/1000;           %division by
1000 to convert N.mm to N.m
end

save ('C:\MobileHome\examples\04_HumanBody\Data\moments.dat',
'moments', '-ASCII');

%-----------------------------------------------------------------------
% create the matrix for the
%    moment = f(ankle)
```

```
moment_function = zeros(s,2);

for i=1:1:s,
   moment_function(i,1) = A(i,17);
   moment_function(i,2) = ( ans.RAnkleMoment(i,1) * Tmass ) /1000;
end

save ('C:\MobileHome\examples\04_HumanBody\Data\moment_function.dat',
'moment_function', '-ASCII');
```

APPENDIX V.B – INTEGRATOR SCRIPT

This script creates the input files used by the *Integrator*.

```
%author: Philippe Ferreira
%date:   1.12.2011

%this script must be used in this way:
%1) define a global variable 'stop' and 'Tmass' with the value get
from the player in Mobile
%            stop  =
%            Tmass =                        (-> this value is the
total mass of the subject)
%2) run the sript
%            integrator
%*****************************************************************
***********

body = load('C:\MobileHome\examples\04_HumanBody\Data\body.dat', '-
ASCII');
n = size(body, 1);              %scalar n = number of samples

dT = 0.01;

%stop >> global variable           >> time when stop was pushed
stop_line = stop*100-1;            % convert time into de number of
the corresponding line (time begin in 0)
stop_line = round(stop_line);      % conversion to an integer

vector = zeros (1, 36);

for i=1:1:36,
 vector(1, i) = body(stop_line, i+1);
end

save
('C:\MobileHome\examples\04_HumanBody\Data\initial_conditions.dat',
'vector', '-ASCII');

matrix = zeros(n-stop_line, 19);        %final matrix

for i=1:1:n-stop_line,                  %insert the data from the c3d
file
    matrix(i, 1)  = dT*i-dT;
    matrix(i, 2)  = body(stop_line+i,11);
    matrix(i, 3)  = body(stop_line+i,12);
    matrix(i, 4)  = body(stop_line+i,13);
    matrix(i, 5)  = body(stop_line+i,14);
    matrix(i, 6)  = body(stop_line+i,15);
```

```
    matrix(i, 7)   = body(stop_line+i,16);
    matrix(i, 8)   = body(stop_line+i,23);
    matrix(i, 9)   = body(stop_line+i,24);
    matrix(i,10)   = body(stop_line+i,25);
    matrix(i,11)   = body(stop_line+i,26);
    matrix(i,12)   = body(stop_line+i,27);
    matrix(i,13)   = body(stop_line+i,28);
    matrix(i,14)   = body(stop_line+i,29);
    matrix(i,15)   = body(stop_line+i,30);
    matrix(i,16)   = body(stop_line+i,31);
    matrix(i,17)   = body(stop_line+i,35);
    matrix(i,18)   = body(stop_line+i,36);
    matrix(i,19)   = body(stop_line+i,37);
end

save ('C:\MobileHome\examples\04_HumanBody\Data\input.dat', 'matrix',
'-ASCII');

%*****************************************************************
***********
moments = load('C:\MobileHome\examples\04_HumanBody\Data\moments.dat',
'-ASCII');
% The moments that came from the c3d file are in N.mm/kg.
% They were already converted in N.m in the readc3d script.
% Now we need to multiply them by the total mass so we'll get the
absolut
% value.

moments_input = zeros(n-stop_line, 2);

for i=1:1:n-stop_line,                  %insert the data from the c3d
file
    moments_input(i, 1)   = dT*i-dT;
    moments_input(i, 2)   = moments(stop_line+i,1)*Tmass;
end

save ('C:\MobileHome\examples\04_HumanBody\Data\moments_input.dat',
'moments_input', '-ASCII');
```

APPENDIX V.C – FINAL MATRIX SCHEME

In Tab. V.1, the scheme of the matrix obtained using the MATLAB script in Appendix V.A is showed. The linear and angular variables were calculated and differentiated in order to obtain the velocities and accelerations. The labels and the units are available on Tab V.2.

Table V.1 Input matrix used to drive all the variables in the *Reader* application

time	trX	. trX	.. trX	trX	. trY	.. trY	rotZ	. rotZ	.. rotZ	…
										…
…	ang1	. ang1	.. ang1	ang2	. ang2	.. ang2	ang3	. ang3	.. ang3	…
…										…
…	ang4	. ang4	.. ang4	ang5	. ang5	.. ang5	ang6	. ang6	.. ang6	…
…										…
…	ang7	. ang7	.. ang7	ang8	. ang8	.. ang8	angHAT	. angHAT	.. angHAT	
…										

In Tab. V.1, the variables of the MBS model are described using the nomenclature used on the scripts. The identification of each variable is included on Tab. V.2.

Table V.2 Identification of the

Variables	Definition	Units
time	Time vector	s
trX	Translation in the x-direction	m
trY	Translation in the y-direction	m
rotZ	Rotation of the hip	rad
ang1	Thigh angle (right leg)	rad
ang2	Knee angle (right leg)	rad
ang3	Ankle angle (right leg)	rad
ang4	Metatarsal angle (right leg)	rad
ang5	Thigh angle (left leg)	rad
ang6	Knee angle (left leg)	rad
ang7	Ankle angle (left leg)	rad
ang8	Metatarsal angle (left leg)	rad
angHAT	HAT angle	rad

In the forward dynamics simulation, some joints were 'released'. With the purpose of adding three DOF at the hip, the joints corresponding to trX, trY and rotZ were set free. The right ankle joint (ang3) was also released, and despite the spring-dampers added at the metatarsal joints, the variables ang4 and ang8 had to be integrated. Thus, the input matrix used in the *Integrator* (see Tab. V.3) is smaller than the one used for the *Reader* (see Tab. V.1) which had all the variables driven.

Table V.3 Input matrix used for the forward dynamics simulation, in the *Integrator* application

time	ang1	ang1	ang1	ang2	ang2	ang2	ang5	ang5	ang5	…
										…

…	ang6	ang6	ang6	ang7	ang7	ang7	angHAT	angHAT	angHAT
…									

APPENDIX VI

INTEGRATOR.CPP

```
#include <Mobile/MoBase.h>
#include <Mobile/MoMapChain.h>
#include <Mobile/MoElementaryJoint.h>
#include <Mobile/MoRigidLink.h>
#include <Mobile/MoMassElement.h>
#include <Mobile/MoLinearSpringDamper.h>
#include <Mobile/MoMechanicalSystem.h>
#include <Mobile/MoLsodarIntegrator.h>
#include <Mobile/MoAdamsIntegrator.h>
#include <Mobile/MoDynamicSystemList.h>

#include <MoUtilities/MoInterpolatingVarListInput.h>
#include <MoUtilities/MoVarListOutput.h>

#include <MoImpactModeling/MoRegImpSpherePlane.h>

#include <Mobile/Inventor/MoScene.h>
#include <Mobile/Inventor/MoWidget.h>

#include <fstream>

#include <Mobile/MoConstantWrench.h>
#include <MoImpactModeling/MoFreeRigidBody.h>


int main () {

/*************************************************************************
*                    DEFINITION OF ALL ELEMENTS                          *
*************************************************************************/
/*    (1) Definition of mechanical system
*************************************************************************/
MoFrame K0, K1, K2, K3, K4, K5, K6, K7, K8, K9, K10, K11, K12, K13,
K14, K15, K16, K17, K18, K19, K20, K21, K22, K23, K24;
MoAngularVariable rotZ('rotZ'), ang1, ang2, ang3('ang3'), ang4, ang5,
ang6, ang7('ang7'), ang8;
MoLinearVariable trX('trX'), trY('trY');
MoVector l_virt1, l_virt2, l_virt3, l_HAT,  l_R_thigh, l_R_shank,
l_R_foot[2], l_R_toes, l_L_thigh, l_L_shank, l_L_foot[2], l_L_toes;
MoFrameList R_K_Outs, L_K_Outs;
R_K_Outs << K12 << K13;
L_K_Outs << K21 << K22;

//Joints
MoElementaryJoint xTrans      (K0 , K1, trX, xAxis)    ;
MoElementaryJoint yTrans      (K2 , K3, trY, yAxis)    ;
MoElementaryJoint Rot         (K4 , K5, rotZ, zAxis)   ;
MoElementaryJoint R_hip       (K6 , K7, ang1, zAxis)   ;
MoElementaryJoint R_knee      (K8 , K9, ang2, zAxis)   ;
```

```
MoElementaryJoint R_ankle     (K10, K11, ang3, zAxis) ;
MoElementaryJoint R_toes      (K13, K14, ang4, zAxis) ;
MoElementaryJoint L_hip       (K6 , K16, ang5, zAxis)  ;
MoElementaryJoint L_knee      (K17, K18, ang6, zAxis) ;
MoElementaryJoint L_ankle     (K19, K20, ang7, zAxis) ;
MoElementaryJoint L_toes      (K22, K23, ang8, zAxis) ;

//Links
MoRigidLink virt1             (K1, K2, l_virt1) ;
MoRigidLink virt2             (K3, K4, l_virt2) ;
MoRigidLink virt3             (K5, K6, l_virt2) ;
MoRigidLink R_thigh           (K7, K8, l_R_thigh) ;
MoRigidLink R_shank           (K9, K10, l_R_shank) ;
MoRigidLink R_foot            (K11, R_K_Outs, l_R_foot) ;
MoRigidLink R_toe             (K14, K15, l_R_toes) ;
MoRigidLink L_thigh           (K16, K17, l_L_thigh) ;
MoRigidLink L_shank           (K18, K19, l_L_shank) ;
MoRigidLink L_foot            (K20, L_K_Outs, l_L_foot) ;
MoRigidLink L_toe             (K23, K24, l_L_toes) ;

//Masses values and inertial tensors
MoReal m_HAT, m_R_thigh, m_R_shank, m_R_foot, m_R_toes, m_L_thigh,
m_L_shank, m_L_foot, m_L_toes;
MoInertiaTensor T_HAT, T_R_thigh, T_R_shank    , T_R_foot, T_R_toes ,
T_L_thigh, T_L_shank, T_L_foot, T_L_toes;

//Offset for dislocating CM
MoVector    offset_HAT,
            offset_R_thigh, offset_R_shank, offset_R_foot,
offset_R_toes,
            offset_L_thigh, offset_L_shank, offset_L_foot,
offset_L_toes;

offset_HAT = offset_R_thigh = offset_R_shank = offset_R_foot =
offset_R_toes = offset_L_thigh = offset_L_shank = offset_L_foot =
offset_L_toes = MoNullState;

//Mass elements
MoMassElement mass_HAT        (K6, m_HAT, T_HAT, offset_HAT);
MoMassElement mass_R_thigh    (K7, m_R_thigh, T_R_thigh,
offset_R_thigh);
MoMassElement mass_R_shank    (K9, m_R_shank, T_R_shank,
offset_R_shank);
MoMassElement mass_R_foot     (K11, m_R_foot, T_R_foot,
offset_R_foot);
MoMassElement mass_R_toes     (K14, m_R_toes, offset_R_toes);
MoMassElement mass_L_thigh    (K16, m_L_thigh, T_L_thigh,
offset_L_thigh);
MoMassElement mass_L_shank    (K18, m_L_shank, T_L_shank,
offset_L_shank);
```

```
MoMassElement mass_L_foot     (K20, m_L_foot, T_L_foot,
offset_L_foot);
MoMassElement mass_L_toes     (K23, m_L_toes, offset_L_toes);

/*    (2) Adding the springs
*****************************************************************/
MoFrame K12_1, K13_1, K15_1, K21_1, K22_1,  K24_1;
MoLinearVariable Tr_1, Tr_2, Tr_3, Tr_4, Tr_5, Tr_6;
Tr_1.q = Tr_2.q = Tr_3.q = Tr_4.q = Tr_5.q = Tr_6.q = 0.0 ;

// geu 2012-01-26
MoReal mass_trick_1, mass_trick_2, mass_trick_3, mass_trick_4,
mass_trick_5, mass_trick_6;
mass_trick_1 = mass_trick_2 = mass_trick_3 = mass_trick_4 =
mass_trick_5 = mass_trick_6 = 1.0;

MoMassElement massElement_trick_1 (K12_1, mass_trick_1);
MoMassElement massElement_trick_2 (K13_1, mass_trick_2);
MoMassElement massElement_trick_3 (K15_1, mass_trick_3);
MoMassElement massElement_trick_4 (K21_1, mass_trick_4);
MoMassElement massElement_trick_5 (K22_1, mass_trick_5);
MoMassElement massElement_trick_6 (K24_1, mass_trick_6);

MoElementaryJoint Trans1 (K12, K12_1, Tr_1, xAxis) ;
MoElementaryJoint Trans2 (K13, K13_1, Tr_2, xAxis) ;
MoElementaryJoint Trans3 (K15, K15_1, Tr_3, xAxis) ;
MoElementaryJoint Trans4 (K21, K21_1, Tr_4, xAxis) ;
MoElementaryJoint Trans5 (K22, K22_1, Tr_5, xAxis) ;
MoElementaryJoint Trans6 (K24, K24_1, Tr_6, xAxis) ;

MoReal k , c ;
MoLinearSpringDamper springDamper1 ( Trans1 , k , c ) ;
MoLinearSpringDamper springDamper2 ( Trans2 , k , c ) ;
MoLinearSpringDamper springDamper3 ( Trans3 , k , c ) ;
MoLinearSpringDamper springDamper4 ( Trans4 , k , c ) ;
MoLinearSpringDamper springDamper5 ( Trans5 , k , c ) ;
MoLinearSpringDamper springDamper6 ( Trans6 , k , c ) ;

/*    (3) Creating the body
*****************************************************************/
MoMapChain Body;
Body << xTrans << virt1
    << yTrans << virt2
    << Rot << virt3
    << R_hip << L_hip
    << R_thigh << L_thigh
    << R_knee << L_knee
    << R_shank << L_shank
    << R_ankle << L_ankle
```

```
    << R_foot << L_foot
    << R_toes << L_toes
    << R_toe << L_toe
    << Trans1 << Trans4
    << Trans2 << Trans5
    << Trans3 << Trans6
    << mass_HAT
    << mass_R_thigh << mass_L_thigh
    << mass_R_shank << mass_L_shank
    << mass_R_foot << mass_L_foot
    << mass_R_toes << mass_L_toes
    << massElement_trick_1 << massElement_trick_2 <<
massElement_trick_3
    << massElement_trick_4 << massElement_trick_5 <<
massElement_trick_6
    << springDamper1 << springDamper4
    << springDamper2 << springDamper5
    << springDamper3 << springDamper6;


//Dynamic equation = variables that will be integrated
MoVariableList  generalizedCoordinates ;
generalizedCoordinates  << rotZ << trX << trY << ang3
                        << Tr_1 << Tr_2 << Tr_3 << Tr_4 << Tr_5 <<
Tr_6;

/*********************************************************************
*              READING THE INPUT FILE for the variables              *
*********************************************************************/
//Variables inserted as input (=drived)
MoVariableList varInputs;
varInputs << ang1 << ang2 << ang4 << ang5 << ang6 << ang7 << ang8;

//name of the input file
string inFileName1('Data/input.dat');

// constructor different than MoVarListInput
MoInterpolatingVarListInput varListInput1(varInputs, OUTPUT_POSITION |
OUTPUT_VELOCITY | OUTPUT_ACCELERATION, inFileName1);
//interpolating because the file gives discontinuous points

MoMapChain inputChain;
inputChain    << varListInput1;

// !!!ALL VARIABLES MUST BE HERE: INTEGRATED IN THE DYNAMIX EQUATION
OR INSERTED AS INPUT
// !!!BE CAREFULL WITH THE NUMBER OF COLUMNS IN THE INPUT FILE
// ANGLES MUST BE IN RADIAN
```

```
/*********************************************************************
*                  READING THE INPUT FILE for the moment             *
*********************************************************************/
//Variables inserted as input (=drived)
MoVariableList momentInput;
momentInput << ang3;

//name of the input file
string inFileName2('Data/moments_input.dat');

// constructor different than MoVarListInput
MoInterpolatingVarListInput varListInput2(momentInput, OUTPUT_FORCE,
inFileName2);
//this list will be added to the integrator

/*********************************************************************
*                        DIMENSIONS & MASSES                         *
*********************************************************************/
/*    (1) Lengths
*********************************************************************/
l_virt1 = l_virt2 = l_virt3 = l_HAT =  l_R_thigh = l_R_shank =
l_R_toes = l_L_thigh = l_L_shank = l_L_toes = MoNullState;
MoReal l_R_foot_x1, l_R_foot_x2, l_R_foot_y, l_L_foot_x1, l_L_foot_x2,
l_L_foot_y;

ifstream input_length ( 'Data/dimensions&masses/lengths.dat' ) ;
input_length >> l_HAT.y
    >> l_R_thigh.y >> l_R_shank.y >> l_R_foot_y >> l_R_foot_x1 >>
l_R_foot_x2 >> l_R_toes.x
    >> l_L_thigh.y >> l_L_shank.y >> l_L_foot_y >> l_L_foot_x1 >>
l_L_foot_x2 >> l_L_toes.x;
input_length.close () ;
l_R_foot[0] = MoVector ( l_R_foot_x1 , l_R_foot_y , 0 );
l_R_foot[1] = MoVector ( l_R_foot_x2 , l_R_foot_y , 0 );
l_L_foot[0] = MoVector ( l_L_foot_x1 , l_L_foot_y , 0 );
l_L_foot[1] = MoVector ( l_L_foot_x2 , l_L_foot_y , 0 );

/*    (2) Masses of all links
*********************************************************************/
ifstream input_masses ( 'Data/dimensions&masses/masses.dat' ) ;
input_masses >> m_HAT
    >> m_R_thigh >> m_R_shank >> m_R_foot >> m_R_toes
    >> m_L_thigh >> m_L_shank >> m_L_foot >> m_L_toes;
input_masses.close () ;

/*    (3) Offsets for dislocating CM
*********************************************************************/
MoReal      off_R_HAT,
        off_R_thigh, off_R_shank, off_R_foot_x, off_R_foot_y,
off_R_toes,
```

```
        off_L_thigh, off_L_shank, off_L_foot_x, off_L_foot_y,
off_L_toes;

ifstream input_offset ( 'Data/dimensions&masses/offset.dat' ) ;
input_offset >> off_R_HAT
    >>      off_R_thigh >> off_R_shank >> off_R_foot_x >> off_R_foot_y
>> off_R_toes
    >>      off_L_thigh >> off_L_shank >> off_L_foot_x >> off_L_foot_y
>> off_L_toes;
input_offset.close () ;

offset_HAT                    = -l_HAT          *     off_R_HAT;
offset_R_thigh                = l_R_thigh       *     off_R_thigh;
offset_R_shank                = l_R_shank       *     off_R_shank;
offset_R_foot                 = MoVector (l_R_foot_x2 * off_R_foot_x,
      l_R_foot_y * off_R_foot_y,    0);
offset_R_toes                 = l_R_toes        *     off_R_toes;
offset_L_thigh                = l_L_thigh       *     off_L_thigh;
offset_L_shank                = l_L_shank       *     off_L_shank;
offset_L_foot                 = MoVector (l_L_foot_x2 * off_L_foot_x,
      l_L_foot_y * off_L_foot_y,    0);
offset_L_toes                 = l_L_toes        *     off_L_toes;

/*    (4) Moment of inertia
**********************************************************************/
MoReal tensor_hat, tensor_R_thigh, tensor_R_shank, tensor_R_foot,
tensor_R_toes, tensor_L_thigh, tensor_L_shank, tensor_L_foot,
tensor_L_toes;

ifstream input_inertia ( 'Data/dimensions&masses/inertia.dat' ) ;
input_inertia >> tensor_hat
    >> tensor_R_thigh >> tensor_R_shank >> tensor_R_foot >>
tensor_R_toes
    >> tensor_L_thigh >> tensor_L_shank >> tensor_L_foot >>
tensor_L_toes;
input_inertia.close();

T_HAT             =       MoInertiaTensor( 0 , 0 , tensor_hat)
      ;
T_R_thigh         =       MoInertiaTensor( 0 , 0 , tensor_R_thigh )
      ;
T_R_shank         =       MoInertiaTensor( 0 , 0 , tensor_R_shank )
      ;
T_R_foot          =       MoInertiaTensor( 0 , 0 , tensor_R_foot )
      ;
T_R_toes          =       MoInertiaTensor( 0 , 0 , tensor_R_toes )
      ;
T_L_thigh         =       MoInertiaTensor( 0 , 0 , tensor_L_thigh )
      ;
T_L_shank         =       MoInertiaTensor( 0 , 0 , tensor_L_shank )
      ;
```

```
T_L_foot          =       MoInertiaTensor( 0 , 0 , tensor_L_foot )
      ;
T_L_toes          =       MoInertiaTensor( 0 , 0 , tensor_L_toes )
      ;

/*********************************************************************
*                        Spring-Damper constants                     *
*********************************************************************/
ifstream input_spring_damper ( 'Data/spring_damper.dat' ) ;
input_spring_damper >> k >> c;
input_spring_damper.close();

/*********************************************************************
*             READING THE INITIAL DATA FROM INPUT FILE               *
*********************************************************************/
//Initial angles and positions
ifstream input_initial ( 'Data/initial_conditions.dat' ) ;
input_initial >>
    trX.q  >> trX.qd  >> trX.qdd  >>
    trY.q  >> trY.qd  >> trY.qdd  >>
    rotZ.q >> rotZ.qd >> rotZ.qdd >>
    ang1.q >> ang1.qd >> ang1.qdd >>
    ang2.q >> ang2.qd >> ang2.qdd >>
    ang3.q >> ang3.qd >> ang3.qdd >>
    ang4.q >> ang4.qd >> ang4.qdd >>
    ang5.q >> ang5.qd >> ang5.qdd >>
    ang6.q >> ang6.qd >> ang6.qdd >>
    ang7.q >> ang7.qd >> ang7.qdd >>
    ang8.q >> ang8.qd >> ang8.qdd ;
input_initial.close () ;

Body.doMotion(DO_POSITION) ;

/*********************************************************************
*                             CONTACT MODEL                          *
*********************************************************************/
MoReal   r1, r2, r3, r4, r5, r6;

ifstream input_radius ( 'Data/contact_model/spheres.dat');
input_radius >> r1 >> r2 >> r3 >> r4 >> r5 >> r6 ;
input_radius.close();

/*    (1) Impact Coefficients readed from a single file
*********************************************************************/
MoRegImpCoeffList ImpCoeff1 = MoRegImp::defaultCoeff;
MoRegImpCoeffList ImpCoeff2 = MoRegImp::defaultCoeff;
MoRegImpCoeffList ImpCoeff3 = MoRegImp::defaultCoeff;

ifstream input_contact ( 'Data/contact_model/3Spheres.dat' ) ;
```

```
input_contact   >> ImpCoeff1.cN                // spring constant in
normal
                                               direction
                >> ImpCoeff1.cT                // spring constant in
tangential
                                               direction
                >> ImpCoeff1.eN                // coefficient of
restitution in
                                               normal direction
                >> ImpCoeff1.eT                // coefficient of
restitution in
                                               tangential direction
                >> ImpCoeff1.exp
                >> ImpCoeff1.mu_st             // sticking friction
coefficient
                >> ImpCoeff1.mu_sl             // sliding friction
coefficient
                >> ImpCoeff1.e_roll
                >> ImpCoeff1.mu_spin
                >> ImpCoeff1.damp_flat

                >> ImpCoeff2.cN                // spring constant in
normal
                                               direction
                >> ImpCoeff2.cT                // spring constant in
tangential
                                               direction
                >> ImpCoeff2.eN                // coefficient of
restitution in
                                               normal direction
                >> ImpCoeff2.eT                // coefficient of
restitution in
                                               tangential direction
                >> ImpCoeff2.exp
                >> ImpCoeff2.mu_st             // sticking friction
coefficient
                >> ImpCoeff2.mu_sl             // sliding friction
coefficient
                >> ImpCoeff2.e_roll
                >> ImpCoeff2.mu_spin
                >> ImpCoeff2.damp_flat

                >> ImpCoeff3.cN                // spring constant in
normal
                                               direction
                >> ImpCoeff3.cT                // spring constant in
tangential
                                               direction
                >> ImpCoeff3.eN                // coefficient of
restitution in
                                               normal direction
```

```
                >> ImpCoeff3.eT                 // coefficient of
restitution in
                                                tangential direction
                >> ImpCoeff3.exp
                >> ImpCoeff3.mu_st              // sticking friction
coefficient
                >> ImpCoeff3.mu_sl              // sliding friction
coefficient
                >> ImpCoeff3.e_roll
                >> ImpCoeff3.mu_spin
                >> ImpCoeff3.damp_flat
                ;
input_contact.close () ;

/*********************************************************************/

MoFrame Kplane;
MoRotationMatrix dR = MoXRotationMatrix(-90.0 * DEG_TO_RAD);
MoRigidLink rot_plane (K0, Kplane, dR);

MoRegImpSpherePlane   R_heel_contact, R_toe_contact, R_tip_contact;
MoRegImpSpherePlane       L_heel_contact, L_toe_contact, L_tip_contact;
R_heel_contact.init   (Kplane, K12_1, r1, ImpCoeff1,
'R_heel_contact');
R_toe_contact.init    (Kplane, K13_1, r2, ImpCoeff2, 'R_toe_contact'
);
R_tip_contact.init    (Kplane, K15_1, r3, ImpCoeff3, 'R_tip_contact'
);
L_heel_contact.init   (Kplane, K21_1, r4, ImpCoeff1,
'L_heel_contact');
L_toe_contact.init    (Kplane, K22_1, r5, ImpCoeff2, 'L_toe_contact'
);
L_tip_contact.init    (Kplane, K24_1, r6, ImpCoeff3, 'L_tip_contact'
);

Body << rot_plane;

/*********************************************************************
*                            INTEGRATOR                              *
*********************************************************************/
MoMapChain integratorChain;
integratorChain << Body;

//Numerical integrator
MoMechanicalSystem  mechanicalSystem ( generalizedCoordinates,
varInputs, integratorChain , K0 , yAxis ) ;

MoDynamicSystemList completeSystem;

completeSystem << mechanicalSystem
//de-activate the contact model by marking the next line as a comment
```

```
      << R_heel_contact << R_toe_contact << R_tip_contact <<
L_heel_contact <<
      L_toe_contact << L_tip_contact
      ;

MoLsodarIntegrator  dynamicMotion ( completeSystem ); //works with
events
MoReal dT = 1.0e-2 ;
MoReal reltol = 1.0e-5 ;
MoReal abstol = 1.0e-4 ;
dynamicMotion.setTimeInterval(dT) ;
dynamicMotion.setRelativeTolerance(reltol) ;
dynamicMotion.setAbsoluteTolerance(abstol) ;

MoMapChain animationChain;                          //new MoMapChain
object
animationChain << inputChain << dynamicMotion;      //add the
integrated model and the output list to the new MoMapChain object

/*********************************************************************
*                     CREATING THE OUTPUT FILE                       *
*********************************************************************/
ofstream fileStream;                        //new output file
string outFileName('Data/output.dat');      //name of the output file
fileStream.open(outFileName.c_str());       //open the output file

MoVariableList outputs;                     //creating the list of
outputs
outputs << ang3;                            //adding every variables I
want to the
                                            output list

MoVarListOutput varListOutput(outputs , OUTPUT_POSITION |
OUTPUT_VELOCITY | OUTPUT_ACCELERATION, fileStream);       //creating
a list of outputs with the variables inserted in outputs

if(fileStream.is_open()){                   //if the output file is open
    animationChain << varListOutput;        //add the integrated model
                                            and the output list to the
                                            new MoMapChain object
}
//elsewhere write an error
else{
    cerr << 'WARNING: couldn't open ' << outFileName << endl;
}




/*********************************************************************
```

```
*                          VISUALIZATION                                *
***********************************************************************/

animationChain.doMotion(DO_POSITION);

//Animation
MoScene Scene ( animationChain ) ;  // interface for 3D-rendering

//Scale factors
float scale = 0.2f, frameScale = 0.1f, massScale = 0.1f;
//View links & joints
Scene.makeShape (R_hip, scale, scale, scale) ;
Scene.makeShape (R_knee, scale, scale, scale) ;
Scene.makeShape (R_ankle, scale, scale, scale) ;
Scene.makeShape (R_toes, scale, scale, scale) ;
Scene.makeShape (L_hip, scale, scale, scale) ;
Scene.makeShape (L_knee, scale, scale, scale) ;
Scene.makeShape (L_ankle, scale, scale, scale) ;
Scene.makeShape (L_toes, scale, scale, scale) ;
Scene.makeShape (R_hip, R_thigh, scale, scale, scale) ;
Scene.makeShape (R_knee, R_shank, scale, scale, scale) ;
Scene.makeShape (L_hip, L_thigh, scale, scale, scale) ;
Scene.makeShape (L_knee, L_shank, scale, scale, scale) ;

//View geometries (=feet+floor)
Scene.makeShape(K11, '/Geometry/footGeometryGreen.so', frameScale,
frameScale, frameScale);           //Green for right
Scene.makeShape(K14, '/Geometry/toesGeometryGreen.so', frameScale,
frameScale, frameScale);
Scene.makeShape(K20, '/Geometry/footGeometryRed.so', frameScale,
frameScale, frameScale);             //Red for left
Scene.makeShape(K23, '/Geometry/toesGeometryRed.so', frameScale,
frameScale, frameScale);
Scene.makeShape(Kplane,  '/Geometry/GridGround.iv');

// contact points
Scene.makeShape(K12_1, '/Geometry/sphere.so', r1, r1, r1);
Scene.makeShape(K13_1, '/Geometry/sphere.so', r2, r2, r2);
Scene.makeShape(K15_1, '/Geometry/sphere.so', r3, r3, r3);
Scene.makeShape(K21_1, '/Geometry/sphere.so', r4, r4, r4);
Scene.makeShape(K22_1, '/Geometry/sphere.so', r5, r5, r5);
Scene.makeShape(K24_1, '/Geometry/sphere.so', r6, r6, r6);

//contact forces
Scene.attachForce(K12_1, 0.2f*frameScale, 5.0f*frameScale);
Scene.attachForce(K13_1, 0.2f*frameScale, 5.0f*frameScale);
Scene.attachForce(K15_1, 0.2f*frameScale, 5.0f*frameScale);
Scene.attachForce(K21_1, 0.2f*frameScale, 5.0f*frameScale);
Scene.attachForce(K22_1, 0.2f*frameScale, 5.0f*frameScale);
```

```
Scene.attachForce(K24_1, 0.2f*frameScale, 5.0f*frameScale);

//view frames
Scene.makeShape(K0 ,  'MoFrameGeom.so', frameScale, frameScale,
frameScale);  //show the origin

Scene.makeCamera(K6, 'Hip');        //camera fixing the frame K6

//view masses
Scene.makeShape(mass_HAT,     'MoMassElement.so', massScale);
Scene.makeShape(mass_R_thigh, 'MoMassElement.so', massScale);
Scene.makeShape(mass_R_shank, 'MoMassElement.so', massScale);
Scene.makeShape(mass_R_foot,  'MoMassElement.so', massScale);
Scene.makeShape(mass_R_toes,  'MoMassElement.so', massScale);
Scene.makeShape(mass_L_thigh, 'MoMassElement.so', massScale);
Scene.makeShape(mass_L_shank, 'MoMassElement.so', massScale);
Scene.makeShape(mass_L_shank, 'MoMassElement.so', massScale);
Scene.makeShape(mass_L_foot,  'MoMassElement.so', massScale);
Scene.makeShape(mass_L_toes,  'MoMassElement.so', massScale);

Scene.addAnimationObject ( animationChain ) ;  // animate
Scene.setAnimationIncrement ( dT ) ;           // animate as fast as
possible

/*************************************************************************
*                              WIDGET'S                                  *
*************************************************************************/

/*    widget1
*************************************************************************/
MoWidget widget1(Scene, integratorChain, 'widget1: Time and edit
variables');
widget1.addSeparator();
widget1.addLabel(animationChain.integratorTime->modelTime, 'Time: ');
//widget.addSeparator();
//widget.addLabel(K7.f, 'K7.f');
widget1.addSeparator();

for(int i=0; i<generalizedCoordinates.getSize(); i++){
    if(generalizedCoordinates[i]->getType()==REVOLUTE)
        widget1.addScrollBar(generalizedCoordinates[i]-
>AngularVariable()->q, -180, 180, generalizedCoordinates[i]-
>getName().c_str());
    else
        widget1.addScrollBar(generalizedCoordinates[i]->q, -5, 5,
generalizedCoordinates[i]->getName().c_str());
}


/*    widget2
```

```
**********************************************************************/
MoWidget widget2(Scene, integratorChain, 'widget2: variables');
widget2.addSeparator();
widget2.addLabel(rotZ.q, 'rotZ.q');
widget2.addLabel(rotZ.qd, 'rotZ.qd');
widget2.addLabel(rotZ.qdd, 'rotZ.qdd');
widget2.addSeparator();
widget2.addLabel(trX.q, 'trX.q');
widget2.addLabel(trX.qd, 'trX.qd');
widget2.addLabel(trX.qdd, 'trX.qdd');
widget2.addSeparator();
widget2.addLabel(trY.q, 'trY.q');
widget2.addLabel(trY.qd, 'trY.qd');
widget2.addLabel(trY.qdd, 'trY.qdd');
widget2.addSeparator();
widget2.addLabel(ang1.q, 'ang1.q');
widget2.addLabel(ang1.qd, 'ang1.qd');
widget2.addLabel(ang1.qdd, 'ang1.qdd');
widget2.addSeparator();
widget2.addLabel(ang2.q, 'ang2.q');
widget2.addLabel(ang2.qd, 'ang2.qd');
widget2.addLabel(ang2.qdd, 'ang2.qdd');
widget2.addSeparator();
widget2.addLabel(ang3.q, 'ang3.q');
widget2.addLabel(ang3.qd, 'ang3.qd');
widget2.addLabel(ang3.qdd, 'ang3.qdd');
widget2.addLabel(ang3.Q, 'ang3.Q');
widget2.addSeparator();
widget2.addLabel(ang4.q, 'ang4.q');
widget2.addLabel(ang4.qd, 'ang4.qd');
widget2.addLabel(ang4.qdd, 'ang4.qdd');
widget2.addSeparator();
widget2.addLabel(ang5.q, 'ang5.q');
widget2.addLabel(ang5.qd, 'ang5.qd');
widget2.addLabel(ang5.qdd, 'ang5.qdd');
widget2.addSeparator();
widget2.addLabel(ang6.q, 'ang6.q');
widget2.addLabel(ang6.qd, 'ang6.qd');
widget2.addLabel(ang6.qdd, 'ang6.qdd');
widget2.addSeparator();
widget2.addLabel(ang7.q, 'ang7.q');
widget2.addLabel(ang7.qd, 'ang7.qd');
widget2.addLabel(ang7.qdd, 'ang7.qdd');
widget2.addSeparator();
widget2.addLabel(ang8.q, 'ang8.q');
widget2.addLabel(ang8.qd, 'ang8.qd');
widget2.addLabel(ang8.qdd, 'ang8.qdd');

Scene.show() ;
```

```
MoScene::mainLoop() ;   // move the scene

// closing the fstream file
if(fileStream.is_open()){
    fileStream.close();
}
return 0 ;
}
```