Human-in-the-loop haptic modelling of one-legged locomotion

Master Thesis

Filip Cengic | 1668072
Department of Biomechanics
Institute of Sports Science
Technical University of Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Imperial College London

BALANCE

Filip Cengic
Student-ID: 1668072
Course of studies: Sports and Computer Science

Master Thesis
Topic: Human-in-the-loop haptic modelling of one-legged locomotion

Submitted: February 16th, 2017

Advisor: Dr. Ildar Farkhatdinov

Prof. Dr. André Seyfarth
Institute of Sport Science
Department of Human Science
Technical University Darmstadt
Hochschulstraße 1
64289 Darmstadt
Germany

Solemn Declaration

Thesis Statement pursuant to § 22 paragraph 7 of APB TU Darmstadt

I herewith formally declare that I have written the submitted thesis independently. I did not use any outside support except for the quoted literature and other sources mentioned in the paper. I clearly marked and separately listed all of the literature and all of the other sources, which I employed when producing this academic work, either literally or in content. This thesis has not been handed in or published before in the same or similar form.

In the submitted thesis the written copies and the electronic version are identical in content.

Date: February 16th, 2017 Signature:

_____

ACKNOWLEDGEMENT

First and foremost, I offer my sincerest gratitude to my supervisor Dr. Ildar Farkhatdinov of the Department of Bioengineering at Imperial College London, whose support and guidance made my thesis work possible. During my stay in London, he steered me in the right the direction, he has been actively interested in my work and has always been available to advise me. I am very grateful for his patience, motivation, enthusiasm, and immense knowledge in the field of haptics that, taken together, made him a great mentor.

I would also like to thank Dr. Franck Gonzales who designed (under the supervision of Prof. Etienne Burdet) the lower level controller for the haptic device interface. Without his expertise and willingness to help while providing necessary input, this work could not have been successfully conducted.

I would like to acknowledge the Institute of Sports Science at Technical University of Darmstadt. My graduate experience benefitted greatly from the courses I took, the opportunities I had under Professor André Seyfarth to serve as a teaching assistant, and the high-quality seminars that the department organized.

Finally, I would like to acknowledge friends and family who supported me during the last 6 months. I would like to thank Karim Simon as the reviewer of this thesis, and I am gratefully indebted for his very valuable comments on this thesis.

Table of contents

1. ... Introduction 1

2. ... State of the Art 3

2.1. Spring-Loaded Inverted Pendulum model (*SLIP*)

2.2. *SLIP* Template for Running 5

2.3. Locomotion Stability 7

2.4. Haptic Interactions

3. ....Haptic Control Design 9

3.1. Principles of a Haptic Interface 10

3.2. Role of Feedback 11

3.3. System description 14

3.3.1. The Controller 15

3.3.2. Haptic Robot 17

3.3.3. Computing Machine 19

4. ....Modelling 20

4.1. Main Variables

4.2. Motion Equations 21

4.3. Touchdown Calculation 25

4.4. Termination Conditions 27

5. ....Experimental Simulation 28

5.1. Calibration of the Haptic Device Robot

5.2. Initial Value Set 29

5.3. Predefined Constants

5.4. Enabling Haptics 31

5.5. Integrator 32

5.6. Touchdown Calculation 33

5.7. Spring Force Computation 35

5.9. S-R Flip-Flop Module 38

6. ....Results 40

6.1. General trials – No haptic input by the user

6.1.1. Changing initial horizontal velocity 41

6.1.2. Changing initial height 42

6.2. Trials with external force application 43

6.3. Trials with spring stiffness manipulation 47

6.4. Trials with spring stiffness manipulation and different height 50

7. ....Interpretation & Discussion 53

8. ....Future Work 55

References 58

List of Figures

Figure 1: *A simple visualisation of the SLIP model*. CC BY-SA 4.0. 4
Figure 2: *Subdivision into phases of running (mod. after Arslan, 2008, p. 9)*. 6
Figure 3: *Elements of haptics (mod. after Moussette, 2012)*. 8
Figure 4: *Basic Principles of Haptic Control Design (mod. after Ma, p. 7)*. 9
Figure 5: *Full System Design (Sensimotor loop)*. CC BY-SA 4.0. 12
Figure 6: *Feedback of the SLIP model displayed on the screen*. CC BY-SA 4.0. 14
Figure 7: *General architecture of the controlling unit*. 16
Figure 8: *Haptic System Setup Environment*. CC BY-SA 4.0. 18
Figure 9: *Estimation of the current leg length*. CC BY-SA 4.0. 24
Figure 10: *Declared constants at the beginning of the script*. CC BY-SA 4.0. 30
Figure 11: *Haptic switch for force application / spring stiffness manipulation*. 31
Figure 12: *SLIP System Integrator*. CC BY-SA 4.0. 32
Figure 13: *Calculation of the foot position during stance phase*. CC BY-SA 4.0. 34
Figure 14: *Ground Reaction Force (GRF) and Gravitation*. CC BY-SA 4.0. 35
Figure 15: *Estimating leg (de-/)compression phase*. CC BY-SA 4.0. 36
Figure 16: *S-R Flip-Flop module for transitional event triggering*. 38
Figure 17: *Changing initial velocity conditions without haptic input by the user*. 41
Figure 18: *Changing initial height conditions without haptic input by the user*. 42
Figure 19: *Haptic force application – 5 trials with* $v_x = 2.0$ m/s. 43
Figure 20: *Haptic force application – 5 trials with* $v_x = 2.5$ m/s. 44
Figure 21: *Haptic force application – 5 trials with* $v_x = 3.0$ m/s. 45
Figure 22: *Haptic force application – 5 trials with* $v_x = 4.0$ m/s. 46
Figure 23: *Haptic stiffness manipulation – 5 trials with* $v_x = 2.0$ m/s. 47
Figure 24: *Haptic stiffness manipulation – 5 trials with* $v_x = 2.5$ m/s. 48
Figure 25: *Haptic stiffness manipulation – 5 trials with* $v_x = 3.0$ m/s. 49
Figure 26: *Haptic stiffness manipulation – 5 trials with* $v_x = 4.0$ m/s. 50
Figure 27: *Haptic stiffness manipulation – 5 trials with CoM height* $y_0 = 1.2$ m. 51
Figure 28: *Haptic stiffness manipulation – 5 trials with CoM height* $y_0 = 1.5$ m. 52

List of Tables

Table 1: *Subdivision into phases of running.* _____ 5

Table 2: *Main modelling parameters for SLIP running.* _____ 20

Table 3: *Initial values for the simulation model.* _____ 29

Abstract

Using haptic control in robotic devices requires a high amount of adaptational skills. In order to control the actuation of a mechanical representation of the human leg, the biomechanics of human anatomy and movement need to be fully understood. By using the simplified spring-loaded inverted pendulum model (*SLIP*) approach, individual muscle contributions to each leg segment are cumulated and result in a summarized force applied within a virtual prismatic spring symbolizing the human leg. Experimental simulation trials for running have been recorded and quantified. A one-legged *SLIP* model with variable input parameters has been developed and implemented within a haptic environment for lower limb control. By applying external forces to the point mass or manipulating the spring stiffness in real-time, running gets unstable and the model falls down after a few steps have been made. Under the concept haptic interaction measures, *SLIP* parameters may be manipulated during the ground contact in order to stabilize the running pattern.

1. Introduction

Humans are, by nature social entities with a remarkable capacity to learn a variety of motor skills, ranging from catching a ball at a young age to jumping and landing on a trampoline. The ability to touch and feel can be described by the haptic sense, which is an integral part of our everyday experience that few of us really notice. Even a small, silent vibrational signal of a smartphone, perceived by the user leads to the acknowledging reactions, such as taking the phone and answering the received message with a call or a text message. Within the scope of this work, the main task is to couple a mathematically expressed running model with a haptic interface. Especially in the field of sensimotor learning, human-in-the-loop haptic modelling of locomotion is a recent subject. The main idea is to apply results – gained from this work – to control a robotics interface paradigm for exoskeletons (B.A.L.A.N.C.E, 2013). Since the control of exoskeleton involves continuous interaction with human body, only computational modelling is not sufficient, as it is hard to model human muscle-skeletal activity. This work deals with human-machine interaction using a 1-degree of freedom haptic device controller. Based on the human-in-the-loop approach (*HITL*), a haptic interface is deployed, which provides mechanical input forces to compute and model the locomotion of running.
The approaches shown in this thesis are based on human motor control skills with the aim to learn and stabilize a virtual running model under unstable conditions. Basically, there are many options on how to introduce parameter changes to the *SLIP* running model. In the following, two cases are taken into consideration:

- Applying an external force to the point mass in real-time
- Changing the spring stiffness in real-time

It should be noted, that all forces and parameter changes are applied during stance phase. More information concerning different approaches for stabilizing locomotion patterns can be retrieved in section 8 (Future Work). By doing experiments on two (different) cases, a research hypothesis is formulated stating that:

"*Spring stiffness manipulations in real-time are more effective than applying an external force to the point mass with respect to achieve a stable running pattern.*"

In order to answer this question, first thoughts on the topic of general locomotion must be made. Locomotion is an interdisciplinary field of studies, e.g. the generation of movement starts in the human brain, where somatosensory information is flowing through the

nerves received by the motor neurons located in the muscle. Taking a common example, e.g. trying to catch the train which is departing in 5 minutes and the distance is still too large for walking. In order to get to the train in time, velocity needs to be incremented and the movement pattern changes from walking to running. What looks so easy in practise, can be very complex when it comes to a mathematical description. Also the complex task of synergetic activation accomplished by the neuromuscular system seems to be easy in practise. In order to understand the basic mechanism of human movement strategies, a simplified model needs to be implemented, which is taking the main forces acting on the human body into consideration.

Haptic interaction with the human being can be regarded as a milestone in modern technology of prosthetics and robotics. The previously mentioned (running) model needs to be coupled with a haptic device interface, which has been developed at Imperial College (2017). This interface is an input-output hardware module, such e.g. a joystick, which can be controlled by the user and also provide feedback. In order to understand the basic functionalities of haptics, it is relevant to know the main technical specifications about the haptic control interface which will be explained into detail in section 3.3 (System Description). The main user task is to control this haptic device, i.e. to manipulate a certain parameter of the hardware-coupled spring-loaded inverted pendulum (*SLIP*) model in real-time. Either, the stiffness parameter can be manipulated by the haptic device robot, or an additional force is applied externally to the point mass. Parameter manipulation of the implemented *SLIP* model is updated in real-time by the human user. Thus, the design should be kept as simple as possible in order to make it usable for everyone. This approach is called human-in-the-loop (*HITL*). In this approach, the human user is the man/manipulator in the middle. Multiple feedback signals (visual and tactile force feedback; see 3.2 Role of Feedback) are fed as an input to the user. Based on the feeding input, the user is able to manipulate parameters in real-time, haptically. The *HITL*-approach forms an endless loop in case that the *SLIP* model doesn't fall down. Sensimotor training may lead to task improvement, when it comes to control of the haptic device robot. Since every user is unique, there are multiple solutions for different parameter configurations. When it comes to sensimotor learning tasks, it is important, that a goal must be specified. How to achieve this goal is up to the haptic control skill level of each user.

2. State of the Art

Creating biomimetic systems can be dated back to the 1950s, where master arms used for remote handling of radioactive materials have been developed (Carignan, 2000, p. 2). These interface systems are called bio-mimetic, because they are designed to mimic the sensations of a human operator (Fisch et al., 2007, p. 1f.). Already existing interfaces for human senses of sight and hearing are well developed, but within the last decades these systems have been extended to a third human sense, which is called *haptics*. By employing new materials and drive components, the mass and friction in these devices have been reduced, making the current generation of haptic devices adequate for many applications (Carignan, 2000, p. 2). With further development of force-torque sensors, such as strain gauge and state feedback mechanisms (e.g. accelerometer) have been incorporated in haptic devices.

Within the scope of this work, the haptic device controller (hardware) is coupled with a *SLIP* model (section 2.1). Differentiations are made between simple and complex running models. Complex modelling approaches include more specified variables describing human locomotion into detail, which requires more computational power. In order to avoid confusion by complexity and to make the model executable on devices with smaller computational capacity, the model is kept simple including only necessary variables (4.1. Main Variables).

2.1. Spring-Loaded Inverted Pendulum Model (*SLIP*)

In this section a commonly used model in the biomechanics of human movement (esp. running) is described, which is called spring loaded inverted pendulum model (*SLIP*). The *SLIP* model is inspired by the observation that the musculoskeletal system shows elastic behaviour while jumping or running. Some species of animals (e.g. kangaroo, ostrich, horses) can store up to 70% of the kinetic body energy at the time of landing in specific elastic elements. This stored energy can be used for the take-off (Alexander, 1976), which contributes to an efficient and energy saving movement. Thus, developing a model based on elastic elements (i.e. springs) is an obvious idea.

In general, any physical model should have a close relationship to reality, based on assumptions (elasticity) and observable effects, such as center of mass and ground reaction forces.

Simply said, the *SLIP* model consist of a point mass located at the center of mass (*CoM*), which is working as a pivot during the ground contact phase. When the spring returns to its initial state length, the flight phase begins. This simple approach of describing the dynamics of running remains relevant after more than 25 years of research.

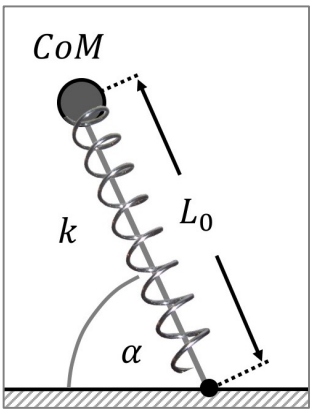

Figure 1: A simple visualisation of the SLIP model

Regarding Fig. 1 the SLIP model can be described by three variables:

- Center of Mass (= point mass)
- Foot position
- Leg function (comparable to a spring)

In the years 1989-1990 Blickhan, McMahon and Cheng mentioned and described the spring-loaded inverted pendulum model (*SLIP*; Fig. 1). The *SLIP* model is inspired by the observation that the musculoskeletal system during running and jumping operates elastically. According to Arslan (2008, p. 8), the spring loaded inverted pendulum (*SLIP*) is the simplest and fundamental template to analyse dynamical locomotion. It was originally used to model running and jumping. There have even been several attempts to extend the model to walking, which is naturally less stable than running (Rossi, 2015, pp. 7ff.). Hence, to reduce complexity of the human musculoskeletal system and to increase stability, the reductive *SLIP* model is used. According to Hutter et al. (2010) this

model is simple, self-stabilising and accurate in predicting the point mass trajectory (Rossi, 2015, p. 7).

2.2. *SLIP* Template for Running

In contrast to walking, both feet are off the ground at some point in time. Respectively, there are two phases (Fig. 2) in one-legged running:

- Flight phase
- Stance phase

Within the flight phase, the only force acting on the *SLIP* running model is the gravitation by the earth. According to Arslan, (2008, pp. 8f.), this phase can be further subdivided (Table 1).

Table 1: Subdivision into phases of running

Flight Phase	*Descent Phase*
	Ascent Phase
Stance Phase	*Compression Phase*
	Decompression Phase

While running, a recurring change between flight and contact phases takes place. In the flight phase, the point mass trajectory is similar to a ballistic curve. When the point mass reached the highest point of the ballistic trajectory (Apex), the vertical velocity changes from positive to negative, this means that the point mass is descending. During stance, the terms *compression* and *decompression* (Fig. 2) play an important role. The leg compression is part of stance phase after the touchdown, here, the rate of leg length change is negative until minimum spring length is reached, i.e. maximum compression of the spring. In this case, the point mass reaches its minimum height, which means that the vertical velocity is positive and the point mass is ascending. Maximum compression of the spring leg is reached, when the center of mass is above the foot. A lot of elastic energy is stored and waiting to be released. The release of energy takes place in the decompression phase, which is a subperiod of stance phase as well. The rate of leg length change is positive and the stored elastic energy of the spring decreases, until the take-off, where the spring returns to its initial state length. The next ground contact is made by the opposite leg.

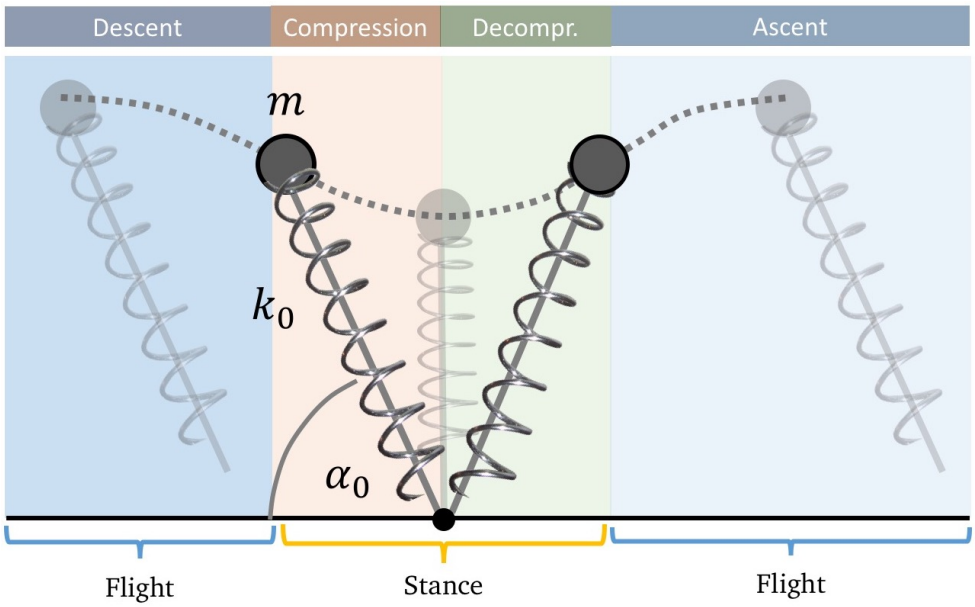

Figure 2: Subdivision into phases of running (in accordance to Arslan, 2008, p. 9)

Transitional events should be considered while running (Arslan, 2008, pp. 10f.), e.g. maximum height or minimum leg length are reached during these events:

1. Apex
2. Touchdown
3. Bottom
4. Take-off

In the following, the upper mentioned events (1-4) will be described in detail. The biomechanical expression *Apex* means, the maximum height of the point mass trajectory by the *SLIP* model is reached. In order to check this event, vertical velocity can be used, i.e. $v_y = 0$. The touchdown event occurs during the descent phase. In this case, the current leg length (L) is equal to the touchdown leg length (L_{TD}), i.e. $L = L_{TD} \ \wedge \ v_y < 0$. At the beginning of stance phase, the touchdown leg length L_{TD} is equal to the leg rest length L_0. During the simulation, the leg compression is calculated, hence, the current leg length L, gets updated ($L < L_0$). Reaching the lowest point of the point mass (center of mass) trajectory is called bottom event. This transitional event occurs between the *compression* and *decompression* subphases. During this event, the spring potential energy reaches its maximum, whereas the leg is at minimum length. Consequently, the rate of change of leg length should be $\dot{L} = 0$. The take-off describes the transition from stance

phase to flight phase and can be expressed as $L = L_{TO} \ \wedge \ v_y > 0$. Here, the current leg length (L) is equal to the take-off leg length (L_{TO}) and the vertical velocity (v_y) increases.

2.3. Locomotion Stability

Stable locomotion is dependent on the initial state and input parameters. As described in 2.1, the *SLIP* model is described by a point mass m that is attached to a massless prismatic spring with its resting length L_0 and stiffness constant k (Hutter et al., 2010, p. 4935). Both, the stiffness constant and the rest length, model the leg, which is carrying the whole body weight. The forces produced by the leg result in measurable ground reaction forces. As the *SLIP* runner is part of a simulation with a predefined starting parameter set and as the surface conditions don't change over time, the result remains the same. To test the model for stability, a success criterion is needed. In contrast to previous works (Seyfarth, Geyer, Günther & Blickhan, 2002), where a *SLIP* model has been developed under stable running conditions, i.e. the initial configuration set is predefined and optimized for the simulation, this work primarily deals with unstable configuration settings. Hence, the goal of stable running is achieved, if the computed *SLIP* model reaches at minimum 5 more steps after the regular fall down. Generally speaking, the *SLIP* model may fall down, if no external forces or stiffness changes were applied by haptic device control.

2.4. Haptic Interactions

Within the scope of this work, the interaction interface between the human wrist and the virtual SLIP model in the scenario of running play a major role. Since the goal of this work is the coupling of a running *SLIP* model with a robotic hardware device, a question might come up concerning the control mechanism of said device. Here, the combination of science and technology, with a well developed feedback design criterion, plays a major role (Moussette, 2012). Research knowledge on haptic device control has already been adopted for various purposes, e.g. Melendez-Calderon et al. (2011, p. 1) from Imperial College London developed a dual-wrist robotic interface, called *Hi5*, which allows the implementation of computer-controlled dynamic conditions and record interaction forces and EMG signals of both partners.

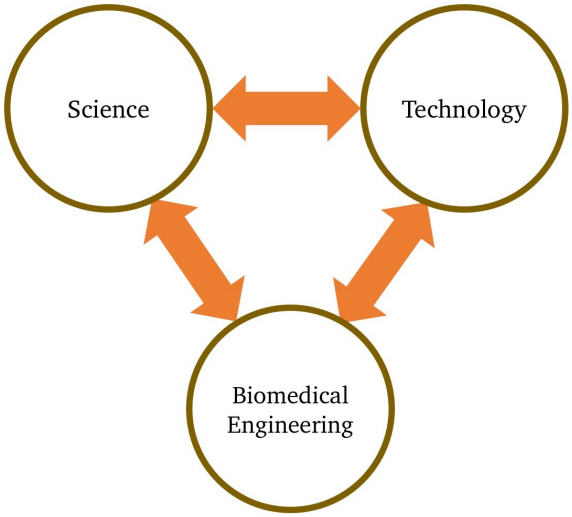

Figure 3: Elements of haptics (mod. after Moussette, 2012)

As it can be seen in Fig. 3, the term *haptics* deals with three main disciplines, i.e. science, technology and biomedical engineering. Science is important, since haptic feedback is perceived by skin receptors. Across synaptic gaps, neural information flows from physiological membranes to the somatosensory cortex. Apart from physiological science, knowledge about the cognitive system of the human brain is necessary, in order to understand the meaning of *haptic interaction*.

Based on (neuro-)scientific results, technological approaches are made and innovative robotic systems could be created. Hence, technology symbolizes practical implementation of scientific facts and biomechanical models. In the context of rehabilitation and assistance in everyday life, technology is responsible for the creation of devices and robotic systems, which are helping people in the execution of individual activities.

3. Haptic Control Design

In order to investigate learning and control of locomotion, a virtual simulation environment is needed (Rossi, 2015, p.13). Within simulation two environments are used: *LabView* (National Instruments) and *MATLAB* (MathWorks) with its built-in Simulink module. For the hardware-coupled interaction with the haptic interface, *LabView* is used. *MATLAB* is needed for finding adequate initial parameters for (un)stable locomotion conditions.

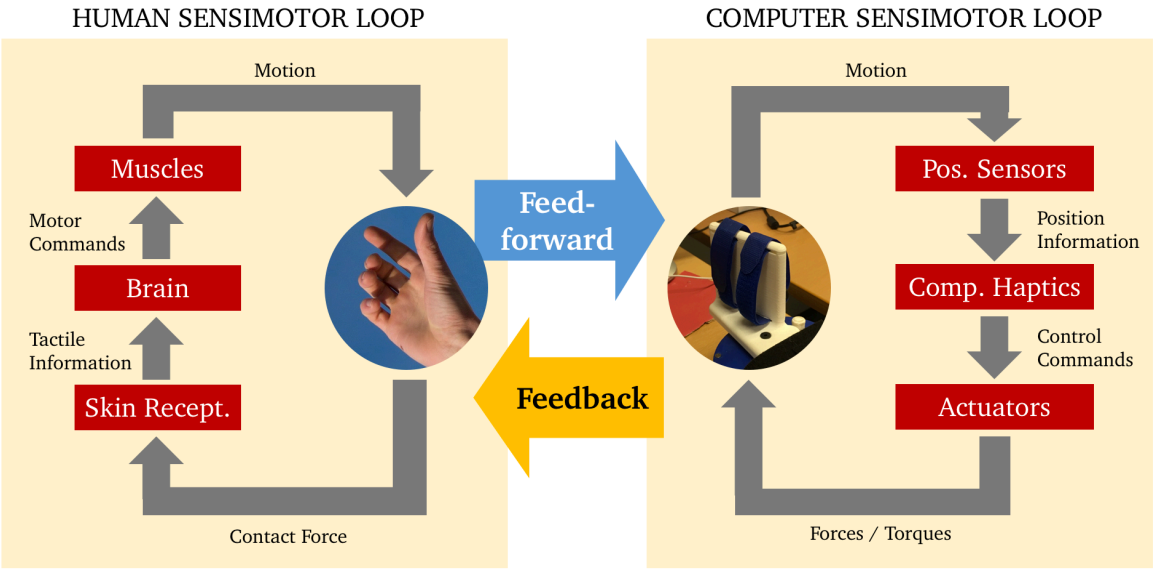

Figure 4: Basic Principles of Haptic Control Design (mod. after Ma, p. 7)

In haptic control design, there are two loops, which are interacting with each other via a virtual environment. The interface for this environment is a hardware controller in form of a joystick. By operating the hardware controller, the contact force feedback is sensed by the human skin receptors (Fig. 4). Providing tactile feedback information to the brain will emerge in new motor commands operated by the neuromuscular system. This paradigm is called Sensimotor learning (Wolpert, 2011). The virtual environment can be regarded as an adaptive control interface with two directions, one for feedback and the other for feedforward.

3.1. Principles of a Haptic Interface

A haptic interface is a sensory-mechanical device, which outputs feedback signals (3.2 Role of Feedback) based on user inputs. The human sense of touch plays a major role when it comes to manipulating simulated object parameters (e.g. the spring stiffness value of a *SLIP* model) in a virtual environment. According to Moussette (2012, p. 40) haptic sense contributes to human perception capabilities along with other sensations, such as vision and audition. The haptic modality is thus a complex bidirectional interface, which encapsulates perception and action like no other sense. Furthermore, the sense of touch comprises a hierarchy of subsystems that include two sensation modules: Tactile and proprioceptive sensation. The tactile module summarizes all sensations arising from an external stimulus to the skin, such as heat, pressure or vibrational effects. Proprioception summarizes internal functionalities of small perception organ units located in muscles, joints and tendons. More information on different feedback channels can be read in the following section 3.2 (Role of Feedback).

Output produced by the haptic device allows the user to feel the (force) feedback coming from the instrumented setup. According to MacLean (2008, p. 150) haptic devices exchange power (e.g., forces, vibrations, heat) through the contact with parts of the user's body, following a programmed interactive algorithm. Actually, haptics surrounds us every day. From a small virtual text message received on a cellular phone, to real sports such as boxing, where each hit against the punching bag causes natural pain reactions, haptics is the key contributor to personal improvements within the scope of sensimotor learning. For basically every kind of learning tasks or self improvement strategies, the human being relies on external feedback.

To implement and understand the basic functioning of haptic process control, the computer must collect data and transmit signals to the controller. In the following, the basic components of a haptic interface will be described in detail:

1. Sensors / Actuators
2. Electronic components
3. Mathematical models

Sensors are one of the useful technologies, which play a major role in the field of haptics (2.6 Haptic Interactions), e.g. for signal measurement and force feedback. Within its mechanical context, a sensor is a transducer that converts a physical stimulus. Sensors can be classified into internal and external sensors. Internal sensors (e.g. accelerometer,

potentiometer, …) are required for basic functioning of the system. External sensors are exposed to the interaction with the outer environment (vision, force, …). As part of the system description in section 3.3 (*System Description*), all relevant sensors of the haptic device environment are listed and specified. To make use of information gained from the sensors, actuation devices are needed. Basically, these are hardware devices, that convert a control signal into a measurable physical parameter (e.g. change of position). Furthermore, a source of energy is required, which may be electric current, hydraulic fluid pressure, or pneumatic pressure. Mathematical models need to be created for the synchronization of the user input and output, respectively (i.e. feedback), as well as the haptic interface.

3.2. Role of Feedback

Haptic interaction is realized by hardware control, which is feeding input to the system, and a visual output of the human-machine interaction is exemplarily displayed on the screen. This simple setup is widely used for sensimotor learning tasks. Feedback plays a major role in the task of sensimotor learning and developing learning skills is even more effective, if every component of the two sensimotor loops (Fig. 4) is well functioning. The usage of haptic device interfaces makes sense, if there is a feedback chain going back to the user, who is controlling the device. Within the scope of this work, sensory integration is a determining factor in the field of haptic control design. According to Rossi (2015, p. 5), sensory integration is very important in stable locomotion as well, because the latter relies on four sensory systems:

- Visual system
- Somatosensory system
- Vestibular system
- Proprioceptive system

The integration of sensory information provided by these systems is necessary to achieve an adult gait pattern (Schmuckler, 1990). Sensory information is redirected via the central nervous system to the human brain. Based on these feedback signals, the human subject is able to improve and adapt its movement patterns. Visual feedback is used to follow the virtual representation of the *SLIP* model. The trajectory of the *CoM* under unstable conditions can be monitored and compared intuitively to regular running

scenarios. Furthermore, visual feedback may deliver a virtual reality experience, for the purposes of rehabilitation and experimental setups. The efficacy of visual and somatosensory feedbacks plays a major role, when it comes to Sensimotor learning tasks. The somatosensory cortex encodes incoming sensory information from receptors all over the body. Based on these signals, new movement programs may be evolved. Tactile perception, e.g. excited by force feedback, is important for the valuation of the input changing parameters (when adding an external force or changing the spring stiffness) fed by the user through the hardware. Vestibular information can be used efficiently, when it comes to balancing tasks or other tasks regarding the maintenance of equilibrium. Considering the scope of this work, vestibular information signals – compared to other sensory system components – are not very relevant. In contrast to vestibular information, the proprioceptive system provides knowledge of body positions, forces and motions via kinaesthetic end organs located in muscles, tendons and joints (Rossi, 2015, pp. 5f.). Within the scope of haptic interface manipulation, proprioception delivers necessary information with regards to the position handling of the haptic assessment tool. Human perceptive sensory organs, such as the muscle spindle or the Golgi tendon organ allow a person with closed eyes, to know where the haptic device robot is located in 3-dimensional space, while someone moves the subjects arms around

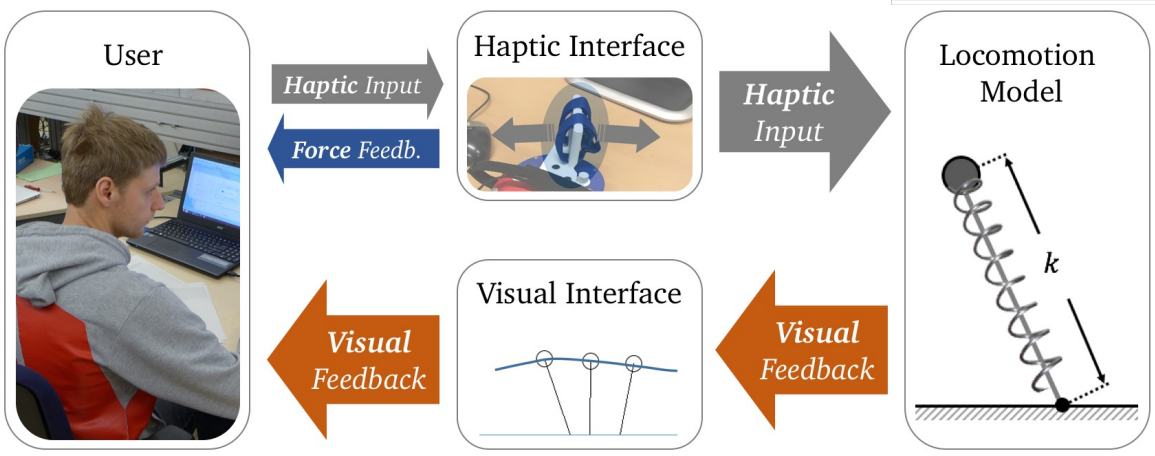

Figure 5: Full System Design (Sensimotor loop)

According to MacLean (2008, p. 149), the ability to touch has evolved to work in a tight partnership with vision and hearing. As shown in Fig. 5, feedback is introduced to the user on two separate channels, i.e. haptically and visually. A force/haptic feedback

mechanism is implemented, which returns, the position of the handle unit to its initial state, in case there is no controller displacement. This means, that all interfacing parameters do not change during the simulation of the implemented *SLIP* model. Haptic feedback is generally divided into two different classes: Kinaesthetic and tactile feedback deal with proprioceptive sensory receptor organs, such as the Golgi tendon organ or the muscle spindle. Latter feedback describes the perceived feeling on the skin surface, which might be caused by external forces/torques applied to the haptic interface controller. Force feedback devices are designed to address proprioceptive sensors in the muscles, joints and tendons by providing forces that react to the wrist flexion/extension movement. These movements are measured, and in response, forces are computed according to a virtual representation of the displayed physical environment (MacLean, 2008, p. 161).

The trajectory of the *CoM* (center of mass) and the leg compression of the *SLIP* model is displayed in real-time. The pivot point, at which the leg is rotating during each stance phase is called center of pressure (*CoP*) and is identical with the foot coordinates of the *SLIP* model. Hence, the dynamics of motion are all calculated during the touchdown. After the take-off only gravitation is acting on the point mass. The visual output of the *SLIP* model is perceived by the user, and since a Human-in-the-loop approach is considered as a major design criterion of the system setup (3.3. System Description), the subject is either able to add an external force acting on the point mass, or to modify the leg stiffness, which is simply modelled by a mechanical spring. Once, the differences in height of the point mass get bigger, it may be an even more challenging task to return into a stable state of the SLIP model.

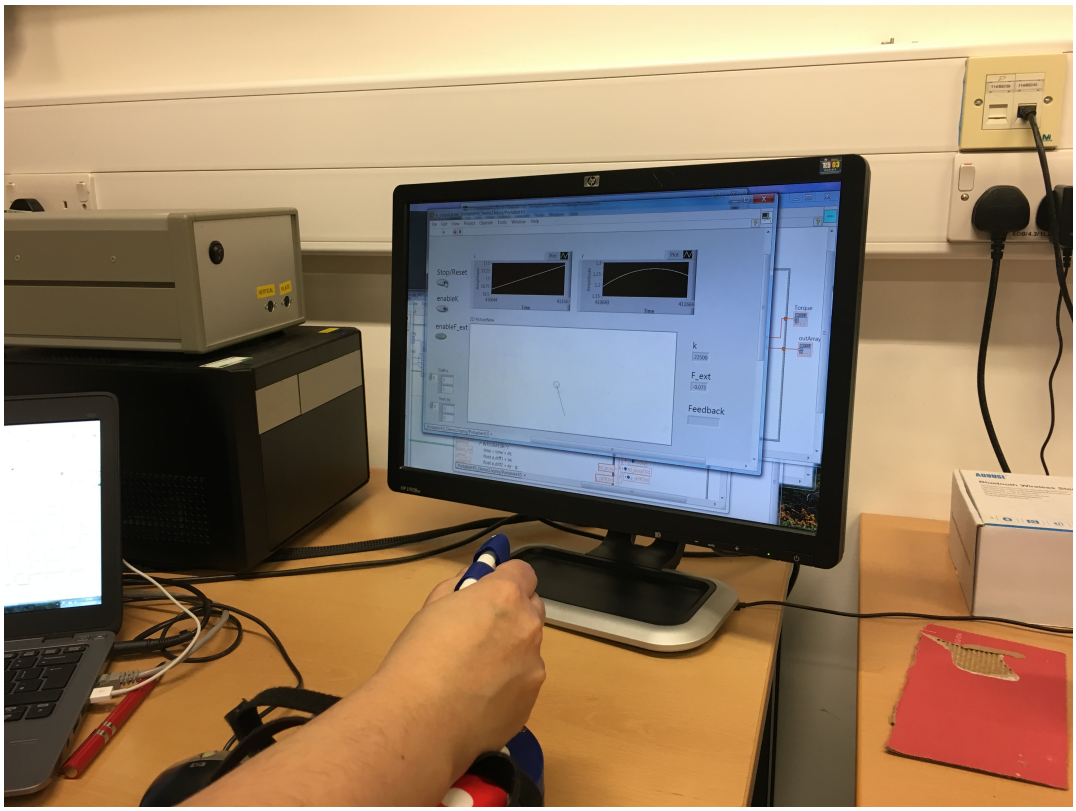

Figure 6: Feedback of the *SLIP* model displayed on the screen

In Fig. 6, the haptic assessment control joystick is shown, which can be used both, as an input and an output device. Depending on the degrees of freedom of the robotic force feedback assessment tool (i.e. haptic interface), model parameters in a computed virtual environment model (e.g. for a *SLIP* runner) can be physically set and even modified in real-time within certain limits such as its workspace and actuator torque. Based on the principles of error-based learning (Wolpert, 2011, pp. 742f. & Krishnapillai, 2015, p. 7f.), experimental trials are repeated 5 times. All 5 trials underlay the same initial parameter set of the experimental setup. Afterwards, another 5 trials are executed with a different start parameter set. Results from all trials with their current initial parameter set are listed and represented graphically in section 6 (Results).

3.3. System Description

In the following, the haptic interface system, called *Hi5*, is described. Due to the design characteristics of the haptic device robot, the interface is easily transportable, allowing the system for easy carry. With a system's workspace of 60° and overall dimensions of 320 x

120 x 250 mm, the system is useable in different spatial environments. According to Wilhelm et al. (2016), the fully grounded *Hi5* interface can be considered as a robot-based tactile assessment tool, due to its force feedback functionality.

The general characteristics of the haptic system are based upon an initial CAD design by Melendez-Calderon et al. (2011). Further readings, regarding the *Hi5* robotic device can be found in the work of Wilhelm et al. (2016). The haptic device can be classified into three different interconnected parts:

- Haptic Robotic Device
- Controller
- Target PC (real-time)

Haptic input is fed into the system through the robotic device controller, which is a customized handle unit connected to a Maxon DC motor (Model RE-65 with 250 W) and a Broadcom encoder (Model HEDR-55L2-BY09) with 3600 counts per turn. This DC motor is among others responsible for the force feedback mechanism (3.2 Role of Feedback) implemented within this system. Encoders are needed for positioning and velocity sensing in a wide range of applications, such as feeding input parameters in real-time into the *SLIP* simulation environment.

As part of the following section, more details will be provided on the hardware and software level side. The actual interface between software and hardware level forms the Controller.

3.3.1. The Controller

The role of the controller is the conversion of a signal coming from the computing machine into motor currents. In the following Fig. 7, plain lines represent power lines and the dotted lines are low current lines for control (dl_1), sensing (q_1) and for the emergency button (EB). The pulse-width modulated (PWM) signal has the symbol dl_1, which stands for drive line and represents the Direction and Enable lines for the driver (Imperial College, 2017, p. 2).

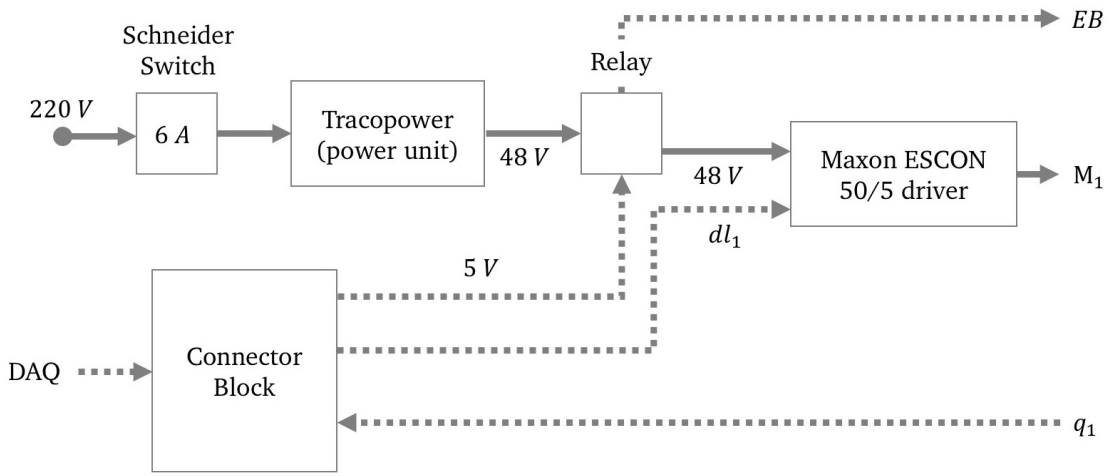

Figure 7: General architecture of the controlling unit (mod. after Imperial College, 2017, p. 2)

As it can be seen above (Fig. 7), the *Hi5* controller embeds a connector block in order to couple the data acquisition card (*DAQ*) with the system. Data acquisition systems convert analogue waveforms into digital values for processing, i.e. signals that measure real world physical conditions are sampled and converted into numeric values, that can be introduced to a computing machine. Furthermore, a power converter with a $48\,V$ output is providing the included driver with a current of up to $3\,A$ in total. The controlling system is using the Maxon ESCON 50/5 plug-in module, because of good control properties and a fast digital current controller with a large bandwidth for optimal motor (current/torque) control. According to the system specification paper by the Imperial College in London (2017, p. 2), the driver is configured to respond to a *PWM* (Pulse-width modulation) and direction signals. The maximum current sent to the motor M_1 is saturated to $3\,A$. Additionally, the fully grounded controller features several electrical safety measures, such as a Schneider electric switch, which makes sure the $220\,V$ line current is below $6\,A$ at all times. A relay can cut the power line between the power converter and the drivers. It is triggered by the facade power switch of the controller, by the emergency stop button and by one of the $5\,V$ lines of the *DAQ* (Imperial College, 2017, p. 2). Hence, drivers can be powered only if the computing machine is turned on. There is a separate Enable line, which is activated by the *DAQ* system (Imperial College, 2017, p. 2).

3.3.2. Haptic Robot

The system consists of a wrist haptic interface fixed to a table on which the subject can place the arm, hold a handle and interact with wrist flexion/extension movements. The haptic device controller is equipped with a Maxon RE-65 DC motor (250 W), that allows the experimenter to program external torques to the wrist joint. Basically, two sensors are necessary in order to record individual torques, i.e. the torque sensor (6.1 *Principles of a Haptic Interface*) and an angular displacement encoder, which can be used for a total angular displacement of 360°. This kind of displacement sensors is concerned with the measurement of the amount by which some object has moved. A torque sensor with a resolution of 1.554 mNm is mounted between the rotating shaft and the handle on each device (Melendez-Calderon, 2011, p. 2579). Considering actuation, a maximum output torque of 0.886 Nm can be reached. Fig. 8 shows the four main components of the system setup; the haptic robot interface is also shown. The position of the handle is recorded in real-time and the tracked handle position and its velocity can be saved by using the degree unit. At the beginning of each experimental trial, the wrist handle returns back to its initial state. But, initial states may vary in accordance to the anatomic properties of the subjects. It is therefore necessary to ensure, that there are no ergonomic differences in the behaviour of serving the handling unit.

This handling unit is driven by rotational movements, that are transmitted to a subject's wrist. According to Farkhatdinov et al. (2015a, p. 5), an important issue is to design a comfortable, adjustable and safe wrist handle. A user should not feel any discomfort during interaction with the device. The orientation of the arm as well as the wrist flexion/extension should remain natural. Even though the workspace for wrist movement is limited, mechanical safety stops are used to restrict the angular range to 60° with a resolution of 0.05°. At software level, safety is guaranteed by limiting the maximum produced torque for the force feedback.

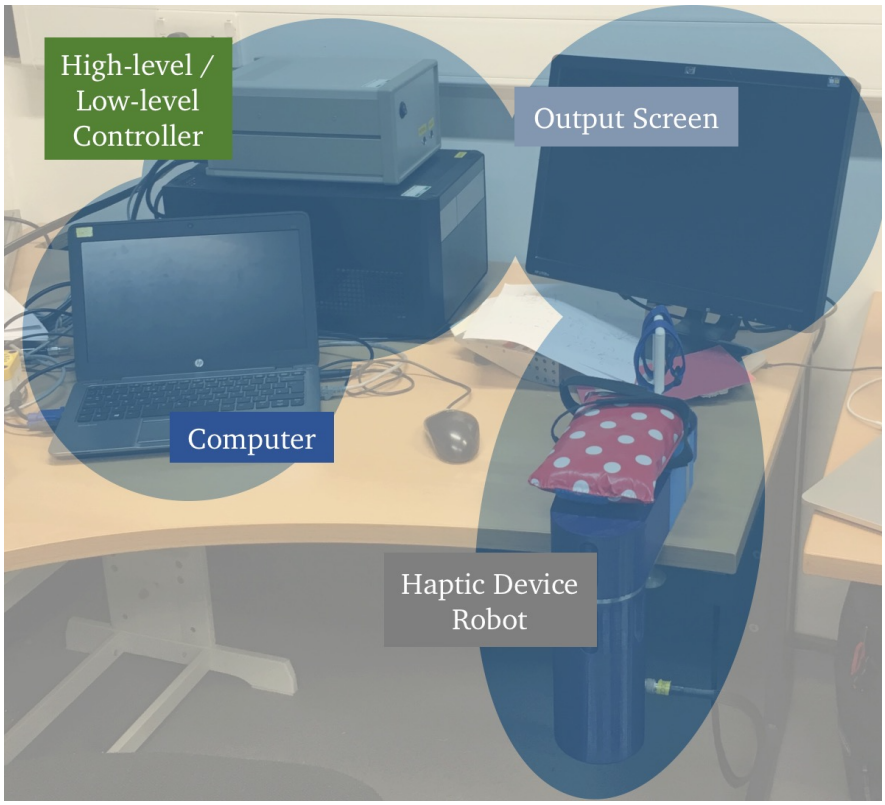

Figure 8: Haptic System Setup Environment

The controller is specific to the robot. Its role is to convert the *PWM* (Pulse-width modulated) signals, which is a modulation technique used to encode data into a pulsing signal, coming from the computing machine. The pulsing signal is generated through motor currents (Imperial College, 2017).

3.3.3. Computing Machine

In the following, differences between the term host and target PC are made. The *SLIP* model is developed and programmed on the host PC, whereas the target PC is responsible for the real-time simulation. Both computing machines are linked by an Ethernet connection. As mentioned in chapter 3.3.1 (The Controller), the target PC features a Data Acquisition (*DAQ*) system (Imperial College, 2017, p. 2).
When implementing software on the host PC, it is important to always work with the same versions of *LabView*. Basically two main engineering software platforms have been used within the scope of this thesis:

- National Instruments *LabView* 2015, Service Pack 1, 15.0.1 (64-bit)
- MathWorks *MATLAB* R2015b (64-bit)

For the development of software programs, which will be coupled with the haptic interface, the *LabView* environment has been used. In the context of software-hardware coupling, *LabView* is more intuitive and user-friendly. Input sources and output sinks can be implemented via drag-and-drop and modified in real-time. All action commands are executed by using a graphical user interface (GUI) on a dedicated computing machine running in real-time. The target PC reads sensor inputs, processes them, and sets the outputs through a data acquisition card under a 1kHz loop.
The *SLIP* model (2.1 Spring-loaded Inverted Pendulum model) has been implemented twice, for *LabView* and for *MATLAB*. *MATLAB* is used once for retrieving stable parameter sets at the beginning of the simulation, and secondly for post-processing tasks. Combining two software applications for diverse implementation purposes and exploiting the benefits of each software, is a key factor for an efficient solution delivery within the scope of this work. For further explanation regarding the software implementation of the *SLIP* model, please have a look at section 5. Likewise, in the experimental setup of Melendez-Calderon (2011), a 22” output screen is used, which allows the presentation of visual feedback to the subject. Additionally, the subject can be provided with information indicating the current wrist position, applied force or movement performance during the task.

4. Modelling

Modelling human motions is a challenging task, especially when it comes to preserve the natural look. In most cases, human motion is a complex sequence consisting of many different submotions. A useful way to solve this problem is to segmentate the motions. An example of the practical benefit of a segmented motion, e.g. is the ability to teach robotic devices how to move like a human being.

4.1. Main Variables

Before the implementation, mathematical models need to be specified in order to make the simulation possible. As already described in 2.1, a *SLIP* model consists of a point mass connected to the ground by a spring, where the spring symbolizes the human leg function. To get the model to function properly, certain variables need to be defined at the beginning of execution. In the following, some conceptual terms are explained in detail, to ensure a basic understanding of this investigation.

Table 2: Main modelling parameters for *SLIP* running

Description	Symbol(s)	Value [unit]
Point mass	m	$80\ kg$
Gravitational acceleration	g	$9{,}81\ \frac{m}{s^2}$
Angle of attack	α_0	$70°$
Leg rest length	L_0	$1\ m$
Current leg length	L	-
Spring stiffness	k	$22.500\ N/m$
Initial conditions	$\begin{pmatrix} v_{x;0} & x_0 \\ v_{y;0} & y_0 \end{pmatrix}$	$\begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$

In order to make modelling easier and more efficient, the whole body mass m is concentrated in one point, which is the center of mass. This point mass is an imaginary point at which the gravitational force acts. Depending on posture and the body mass

distribution, the position of the center of mass may vary. The center of mass is connected by leg to the foot. Beside the point mass, the rest of the system is considered massless. The gravitational acceleration constant g holds the commonly used average value of 9,81 m/s^2. Depending on the distance to the geocenter, this value may change slightly, which will be disregarded within the framework of this model. The angle of attack α describes the orientation of the legs relative to the ground. This angle is set as a fixed parameter and is essential to the further calculation of human movement. While running, the angle α gets higher until the take off starts and a new running step begins.

The leg rest length L_0 plays an important role, when it comes to the simulation of the take-off resp. flight phase. This value is set to 1, which constrains further elongation of the leg. By subtracting the rest length L_0 of the leg from the actual leg length L, the change of the leg length can be estimated. The spring stiffness holds the value of 21.000 N/m. Increasing resp. decreasing this value leads to higher resp. lower force production by leg after each touch down. Values, which are significantly deviating from 21.000 N/m, are resulting in unrealistic model behaviour. Since the running model contains a looping structure with its Euler integration block, previously calculated values need to be fed as input parameters during each iteration. When running the model first time, those input values need to be predefined. In total there are four initial values, two for velocity resp. position in x- and y-direction.

4.2. Motion Equations

Newton's fundamental law of dynamics forms the basis for the following motion equations. According to Newton's 2nd law, the acceleration (symbolized by two dots over the variable) is proportional to the acting force $\vec{F}$ (ME1). The mass m has inertial influence in horizontal and vertical direction of the movement. The proportionality factor corresponds to the mass m to be moved. For the mathematical description of the SLIP runner, vectors are used. Later on, when it comes to the simulation part (chapter 5), scalars are used.

$$\vec{F} = m * \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} \qquad \text{(ME1)}$$

As mentioned in the previous chapter 2.1, the mechanical spring is the basic component of the *SLIP* model. Springs can have linear, as well as progressive and degressive force-length functions. Within the range of this work, the simple linear case is used. Thus, the generated force F_{SPRING} (ME2) is proportional to the difference between the resting length L_0 and the current spring length L. The proportionality factor is the spring stiffness k.

$$F_{SPRING} = -k * (L_0 - L) \quad \text{(ME2)}$$

While running, a recurring change between flight and contact phase takes place. During stance phase (chapter 2.2), one leg is in contact with the ground. The other leg is swinging forward. Therefore, it is sufficient for the determination of the occurring forces to consider only one leg during each running cycle. Furthermore, the movement of running is considered exclusively in the sagittal plane, the analysis of motion can be done in two-dimensional space. In this case occurring forces can be subdivided into horizontal (x-component) and vertical forces (y-component), respectively.
In contrast to walking, where one foot always touches the ground, both feet lift off the ground during the flight phase. Hence, the only force acting on the point mass is gravity. The gravitational force can be described as follows:

$$\vec{F}_G = m * \begin{pmatrix} 0 \\ -g \end{pmatrix} = \begin{pmatrix} 0 \\ -mg \end{pmatrix} \quad \text{(ME3)}$$

For the calculation of the leg force $\vec{F}_{LEG}$ produced by the muscular system during the ground contact phase four values are needed:

1. Spring stiffness k
2. Leg rest length L_0
3. Angle of attack α
4. Current leg length L

The corresponding mathematical expression multiplies the stiffness factor k with the change of leg length. By making use of trigonometric functions, the leg force $\vec{F}_{LEG}$ can be subdivided into a horizontal and a vertical component. Because of easier notation, vectors are used for modeling the SLIP runner within this chapter, although vectors are not permitted for the simulation (chapter 5) of the running model:

$$\vec{F}_{LEG} = -k * (L_0 - L) * \begin{pmatrix} \cos(\alpha) \\ \sin(\alpha) \end{pmatrix} \quad \text{(ME4)}$$

The first three parameters are constants and do not change with time. Only the current leg length L is time dependent and can be estimated by using the following mathematical expression:

$$L = \sqrt{(CoM_x - TD_x)^2 + \left(CoM_y - TD_y\right)^2} \quad \text{(ME5)}$$

$$\Leftrightarrow L = \sqrt{x^2 + y^2} \quad \text{(ME6)}$$

As it can be seen in ME6, the difference between point mass ($CoM_{x;y}$) and touchdown position ($TD_{x;y}$) is abbreviated in this particular case by the symbols x and y. The symbol y is equal to the height of the *CoM*, because the height of the touchdown position TD_y is equal to 0. More information concerning the calculation of the touchdown position ($TD_{x;y}$) can be retrieved in section 4.3 (*Touchdown Calculation*).

With respect to formula ME4, the trigonometric functions at the end of the expression, can be replaced by its corresponding length ratios:

$$\cos(\alpha) = \frac{x}{\sqrt{x^2 + y^2}} = \frac{x}{L} \quad \text{(ME7)}$$

$$\sin(\alpha) = \frac{y}{\sqrt{x^2 + y^2}} = \frac{y}{L} \quad \text{(ME8)}$$

Regarding formula ME5 for the estimation of the current leg length is based on Pythagorean theorem (ME6). In this case, the spring length L is the hypotenuse. The height y of the center of mass (CoM_y) and the distance x between the CoM_x and the touchdown position TD_x (chapter 4.3) represent the two cathetes x and y, respectively (Fig. 9).

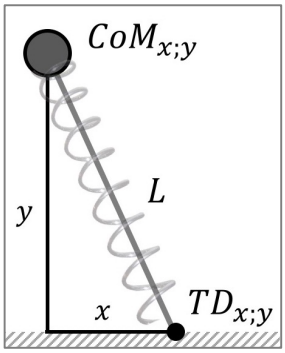

Figure 9: Estimation of the current leg length

During stance phase, the force produced by the leg results from the compression of the spring. The direction of the acting force is in parallel to the spring force. When inserting formulas ME6, ME7 and ME8 into ME4, the leg force can be mathematically expressed as:

$$\vec{F}_{LEG} = -k * \left(\frac{L_0}{\sqrt{x^2 + y^2}} - 1 \right) * \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{(ME8)}$$

The motion equations result during stance phase from the sum of the gravitational $\vec{F}_G$ and the leg force $\vec{F}_{LEG}$:

$$m * \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = \vec{F}_{LEG} + \vec{F}_G \quad \text{(ME9)}$$

$$m * \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = -k * \left(\frac{L_0}{\sqrt{x^2 + y^2}} - 1 \right) * \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ -mg \end{pmatrix} \quad \text{(ME10)}$$

Regarding ME10, it becomes apparent, that the motion equations form a differential equation system, i.e. the equation depends on both, a certain variable and its derivatives. This equation is solved numerically and will be presented in the following chapter 5 (*Experimental Simulation*).

4.3. Touchdown Calculation

Forces are applied to the point mass during stance phase, either externally or internally by changing the spring stiffness parameter. All necessary mathematical operations are calculated as part of the spring leg force production ($\vec{F}_{LEG}$). Basically, the cumulated position of the virtual foot is computed, when it comes to the estimation of the touchdown. By using trigonometric functions, the x-coordinate of the foot can be determined as follows:

$$Pos_x = L_0 * \cos(\alpha_0) \quad \text{(ME11)}$$

$$Pos_y = L_0 * \sin(\alpha_0) \quad \text{(ME12)}$$

As it can be seen in ME11, for the calculation of the touchdown x-position, two constants (L_0, α_0) are needed, which have been specified in section 4.1 (*Main Variables*). The constant variable Pos_x specifies the local x-component of the virtual representation of the human foot, whereas Pos_y is the calculated landing height of the point mass (Fig. 9, *CoM*), depending on the leg rest length L_0 and the angle of attack α_0. The landing height Pos_y doesn't get incremented, unlike the x-component of the global touchdown position TD_x (ME16). The local x-component of the *CoM* can be estimated by a simple mathematical subtraction, i.e. $TD_x - Pos_x$. Regarding this expression, previously estimated Pos_x variable (ME11) is needed, in order to increment the global touchdown position TD_x (ME16), which is equal to the achieved horizontal running distance in x-direction.

As part of the next simulation processing step, the difference is taken between the current point mass (*CoM*) height y and the constant height Pos_y (ME13). It is checked, if the

difference is less than 0. This suggests, that the *SLIP* running model is in stance phase, because the current *CoM* height y is lower, than the constant initial height of the point mass (Pos_y). Since the angle of attack α_0 is a globally defined and constant value, touchdown always takes place with an angle of 70° between the leg axis and the ground line. This leads to a simplified approach for estimating stance and flight phase. Considering the values of the previously computed iteration, it is checked, if the difference between these two point mass heights (ME14) is greater than 0. Mathematically expressed, this would look like $diff_y \leq 0 \ \wedge diff_{y;-1} \geq 0$. This implicates, that the point mass is in descent phase, i.e. falling towards the ground due to the gravitational acceleration.

$$diff_y = y - Pos_y \tag{ME13}$$

$$diff_{y;-1} = y_{-1} * \mathrm{Pos}_{y;-1} \tag{ME14}$$

This transitional event for the touchdown estimation can be determined only once within a running cycle. Furthermore, the parameter $TD_{allowed}$ must be set to 1 ($TD_{allowed} = 1$), otherwise the following three mathematical assignments (ME15-ME17) won't be executed. $TD_{allowed}$ is a redundant variable, which checks, if the *SLIP* running model already touched down during a running cycle. Usually this isn't the case, because only one touchdown per cycle is allowed.

$$r = 1 \tag{ME15}$$

$$TD_x = x + \mathrm{Pos}_x \tag{ME16}$$

$$TD_y = 0 \tag{ME17}$$

Setting the r-value to 1 implicates, that a touchdown has been recorded for the current running cycle (ME15). Hence, at the end of the script block in the following iteration step,

the $TD_{allowed}$ parameter will be set to 0 until the next running cycle begins. Nevertheless, checking if $TD_{allowed} = 1$ is an additional measure, because usually there is one transitional event for the touchdown estimation during a locomotion cycle. In ME16, the running distance (equally to accumulated touchdown position TD_x) gets incremented by the local constant value Pos_x (ME11).

More information regarding modelling and simulation of the *SLIP* runner, can be found in chapter 5 (*Experimental Simulation*).

4.4. Termination Conditions

Because *LabView* only computes values, but doesn't check whether these results are useful, manual termination conditions need to be implemented within the system. The simulation will stop as soon as certain conditions for running are no longer fulfilled. First termination condition results from simple logical reasoning. If the height of point mass crosses the value 0 ($y < 0$), the *SLIP* model has fallen to the ground. Since the *SLIP* running model isn't able to get up, the simulation has to be terminated at this point.

Another possible termination condition is to implement a step counter. Therefore, a success criterion for stable running is needed. If the maximum amount of steps has been reached during a simulation, the model is considered as stable and the simulation can terminate. In order to implement a step counter, the vertical velocity (v_y) of the point mass (*CoM*) is necessary for the computation.

Considering that within the previous iteration step – immediately before Apex (highest point of the *CoM*) height is reached – the respective velocity value is positive ($v_y \geq 0$), after one iteration step the vertical *CoM*-velocity v_y becomes negative ($v_y \leq 0$). In this particular event, the implemented step counter increments itself by the value 1.

5. Experimental Simulation

LabView offers different script blocks. Due to debugging reasons, formula node scripts are used. All input/output ports hold standard symbol names in order to make it easy to follow the execution structure. The advantage is that the code is manageable and easier for correcting errors.

Within this section, the previously described (chapter 4 *Modelling*) continuous motion equation formulas are implemented into a single coding block. The coding block can be subdivided into following main sections:

1. Predefinition of motion equational constants
2. Enabling haptics
3. Integrator
4. Touchdown Calculation
5. Leg – Spring Force Computation
6. *S-R* Flip-Flop module as a trigger

Within the next subchapters, the main coding sections (1-5) will be described into detail.

5.1. Calibration of the Haptic Device Robot

Before recording experimental trials from the haptic device system to the target PC, the robotic assessment tool needs to be calibrated properly. Thanks to the simple, intuitive handling of the software implemented in *LabView*, calibration is done by a single click.

More information on the topic of calibrating the *Hi5* haptic device can be found in recent readings of Imperial College (2017).

5.2. Initial Value Set

In accordance with the modelling section 5.1 (*Main Variables*), following variables are necessary for the simulation (Table 3). Variable names shown in Table 3 are similar to the names set in the *LabView* implementation script. A sample time of $^1/_5$ millisecond ($dt = 0.0002$) is regarded as sufficiently precise for the simulation. For computational reasons, instead of arrays and vectors, scalars are used.

Table 3: Initial values for the simulation model

Initial conditions	Variable	Value
Sample time	dt	0.0002
Point mass	m	80
Gravitational acceleration	g	9.81
CoM x-position	x	0
CoM y-position	y	1
Horizontal velocity	vx	5
Vertical velocity	vy	0
Leg rest length	$L0$	1
Spring stiffness factor	k	22,500
Angle of Attack	$alpha0$	1.22173

It might be confusing, that the unit [m] stands for meter, instead the symbol „m" stands for the point mass. Please consider this, while continuing to read.

5.3. Predefined Constants

As already described in the modelling section (chapter 4), certain initial values need to be predefined before simulation starts (Table 3). These initial conditions are set as global variables within the simulation environment and are accessible in real-time during the simulation.

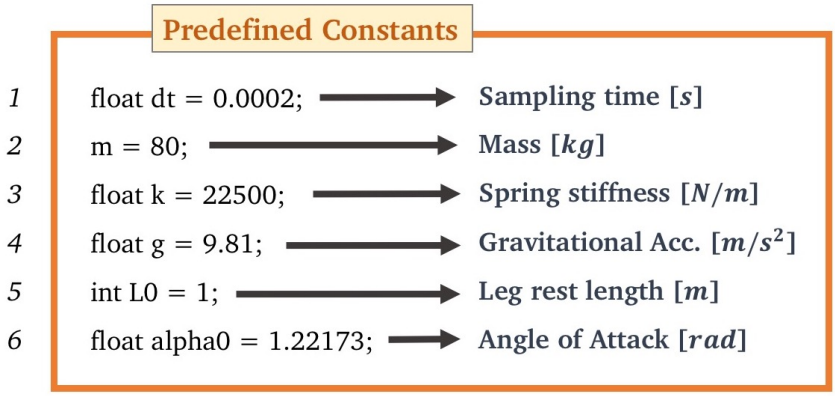

Figure 10: Declared constants at the beginning of the script

As it can be seen in Fig. 10, all constant variables, which are not input parameters, need to be declared before the *LabView* node script is executed. Since the mass m (Fig. 10, line 2) is globally defined (i.e. global variable), there is no declaration. The declaration type *float* means floating point number. Almost all variables within the upper code are of this type, beside the leg rest length parameter $L0$, which can be an integer (abbr. *int*) as well. Therefore, the value 1 has been chosen (4.1 *Main Variables*).

5.4. Enabling Haptics

Since the current position of the calibrated haptic device robot (section 5.1 *Calibration of the haptic device robot*) is fed as an input parameter to the main formula node system block, a switch determines the experimental mode. Concerning the *SLIP* model approach, the following two experimental modes (1, 2) have been introduced:

1. External force application to the *CoM*
2. Changing the spring stiffness k

By using a case structure at upper hierarchical level of the software implementation (parental code), the two parameters *enableK* and *enableF_ext* are set to the values 0 or 1. In case that first and third if-statements down below are true (Fig. 11; lines 1, 3, 5), no experimental mode is selected, which means that the *SLIP* model runs without external manipulation changes via haptic device control.

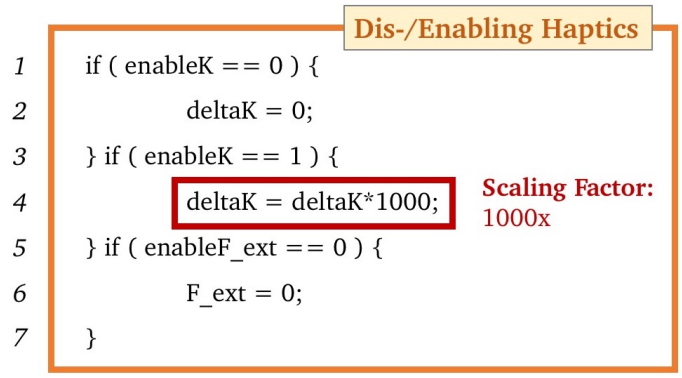

Figure 11: Haptic switch for force application / spring stiffness manipulation

Basically, the two parameters *deltaK* and *F_ext* are proportional to the recorded position of the robotic assessment handle. Due to the widely changing numbers for the spring stiffness (22.500 ±), the variable *deltaK* needs a scaling factor of 1000 is needed (Fig. 13, line 4). Otherwise the external manipulation via the influence of haptic control design would be unremarkably small. Incrementing and decrementing the leg spring stiffness k by values of 1000 N/m leads to major influences with respect to the running pattern of the *SLIP* model.

5.5. Integrator

The integrator forms the core of each iteration step within the *LabView* model and it is used to calculate the current position (x, y) and velocity (vx, vy). Both, the starting speed and the starting position, as well as the gravitational acceleration g are required as input parameters. These input parameters are used for the initial calculation and all subsequent integrations, respectively.

Technically it is sufficient to use five lines of code, in order to estimate the approximated velocities and positions numerically. As there is no horizontal force, resp. acceleration, acting against the *CoM*, the second line of code doesn't need to be considered any further. In contrast, there is the gravitational acceleration g acting permanently on the point mass. Hence, gravitation must be subtracted from the current point mass acceleration (Fig. 12, line 3). Taking the difference in acceleration (a_diff1, a_diff2) and multiplying it with the sample rate dt during each iteration step, velocity can be calculated. Within the lines 8-9 (for *CoM* velocity) and 10-11 (for *CoM* position), the numerical Euler integration approach is used (double integration). Globally defined variables (5.2 *Initial Value Set*) are used as input/output port values.

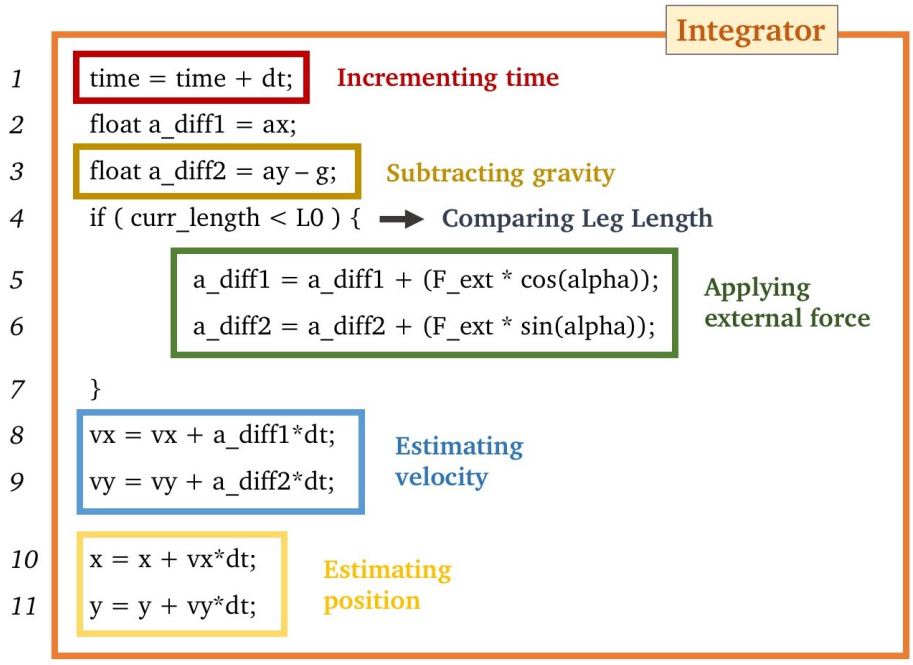

Figure 12: *SLIP* System Integrator

Within the lines 5-6 the current leg axis angle α is calculated, which is the angle between the leg axis and the ground line. In order to calculate the angle α, *LabView*-intern function atan2 is used, which is the arctangent function with two arguments (SIM1).

$$alpha = atan2(y, (abs(td\_posx - x))) \quad \text{(SIM1)}$$

As first argument, the position y of the *CoM* is needed. The second argument is the absolute value of the difference between the cumulated touchdown x-coordinate (td_posx) and the current point mass x-position. According to the laws of trigonometry, the difference ($td\_posx - x$) decreases within each iteration step, while the current angle $alpha$ increases.

In line 4, the current leg length $curr\_length$ is compared to the leg rest length $L0$. If the current length is smaller, this means that the virtual representation of the human leg gets compressed while descending, i.e. the *SLIP* running model is in stance phase. The following two lines 5-6 are embraced by the if-clause, updating the point mass acceleration values in x- and y-direction with respect to the first experimental mode (5.4 *Enabling Haptics*). The first mode deals with haptic device interaction by applying an external force F_ext to the *SLIP* system.

In order that the integrator is able to calculate the velocity and position values for each iteration step, it needs to be defined, when a step is completed and when a new iteration begins. For this purpose, the landing heights must be calculated (chapter 5.6 *Touchdown Calculation*).

5.6. Touchdown Calculation

Since the integration part (5.5 *Integrator*) of the formula node block is responsible for the position calculation of the point mass, further calculations need to be made in order to compute the landing heights of the point mass or the touchdown position of the foot.

The position of the foot td_posx is required when computing locomotion dynamics during ground contact phase (i.e. stance phase) as well as for the generation of spring force (5.8 *Spring Force*). As described in section 4.3 (*Touchdown calculation*), at the beginning of the touchdown calculation, the *CoM* landing heights need to be determined (Fig. 13, line 1-2). For estimating landing height, the inclination angle α of the foot between the

functional leg axis and the ground line, as well as the initial spring length (L_0) are needed as input parameters. These two parameters are predefined as initial conditions (chapter 5.2 *Initial Value Set*) and assumed to be constant for each touchdown.

The first line in the code (Fig. 13, line 1) specifies the local x-coordinate of the virtual representation of the human foot. In order to estimate the landing height, trigonometric functions are used. Since the result is a multiplication of constants (L_0, α_0), the main result is constant as well. In every iteration loop, the difference (y_{diff}) between the current height of the point mass and its initial height at the beginning of each touchdown, is calculated (line 3). This value is compared to the calculated value of the previous iteration. If the difference is negative, the system knows, that the *SLIP* is falling. In line 6, a redundant check by the S-R Flip-Flop is processed to see, whether the model is still running properly.

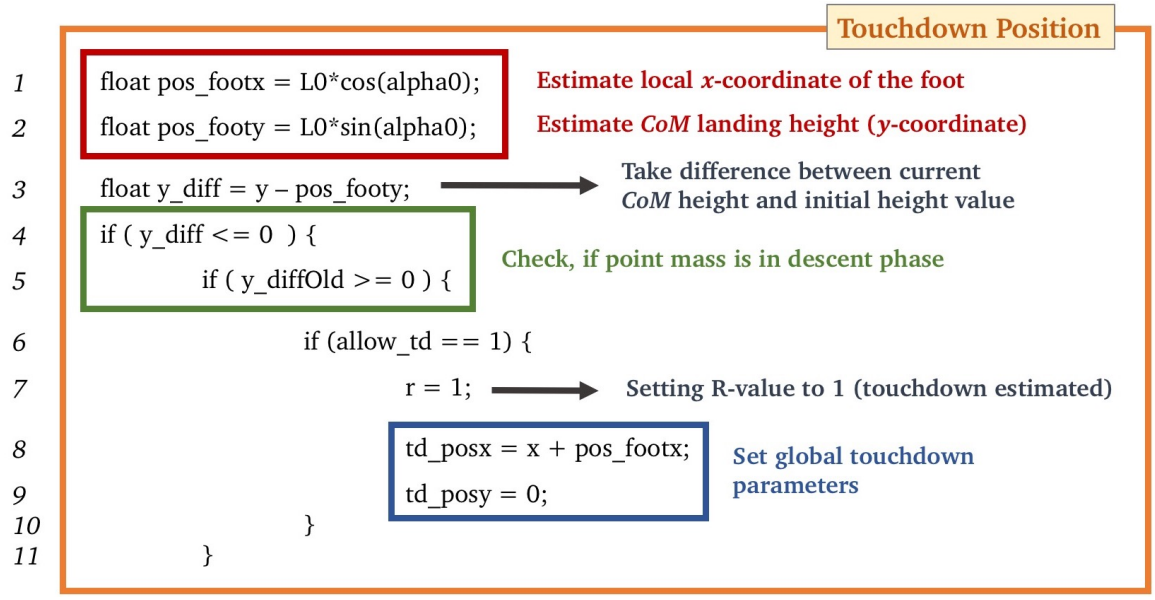

Figure 13: Calculation of the foot position during stance phase

If there is a transitional touchdown event within an iteration loop, the global values for the foot x-/y-coordinates get updated.

5.7. Spring Force Computation

Certain values, such as the *CoM* position and velocity can be derived from the leg function. The mechanical function of the involved biological elements in the human leg (bones, muscles, tendons, connective tissue, sensory-motor system, etc.) can be represented by a spring, which is acting across the leg axis (Fig. 14). According to Newton's 3rd law, the respective ground reaction force (F_{GROUND}, big red arrow) is equivalent to the force exerted by the leg muscles. In contrast to the ground reaction force, which applies during stance phase, the gravitational force (F_G) permanently acts on the point mass. As mentioned in the previous section 5.6 (*Touchdown Calculation*), the ground reaction force (F_{GROUND}) is subdivided into a horizontal (x-direction) and a vertical (y-direction) partial force. By adding these two force vectors, the resultant force F_G can be estimated. Modifying the resultant force F_{GROUND} can be treated as if applying an external force to the point mass (experimental mode 1), with the exception that the externally applied force F_{ext} acts in positive x-direction, i.e. in the direction of running. Both, F_{GROUND} and F_{ext} can be subdivided into a horizontal and a vertical component.

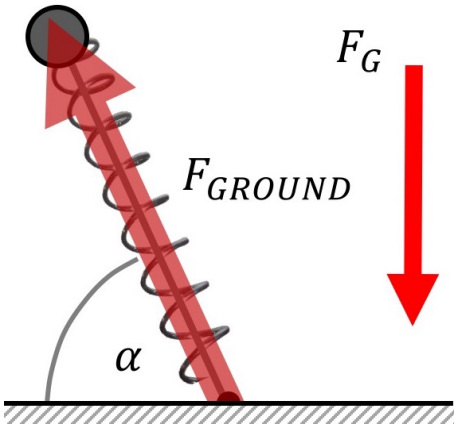

Figure 14: Ground Reaction Force (*GRF*) and Gravitation F_G

In order to calculate the produced spring force generated through ground reaction, certain trigger variables (e.g. *S*-value in Fig. 15, line 10) need to be activated. For the calculation of the current leg length *L* the horizontal distance in *x*-direction of locomotion between the *CoM* and the touchdown position is needed (line 1). Since the value *td_posy* holds the value 0 during stance phase, the point mass height can be described by the *y*-value of the *CoM* (line 2). According to Pythagorean theorem (chapter 4.2, ME6), the *SLIP* representation of the human leg (hypotenuse) forms a triangle with the ground line and the virtual orthogonal line connected to the *CoM*. The Euclidean distance of the two cathetes is squared, summed up and afterwards the square root is taken (Fig. 15, line 3).
As part of this thesis, experimental trials are based on the human-in-the-loop (*HITL*) approach. Therefore, the stiffness factor *k* might change during the real-time simulation, depending on the position scalar *deltaK* fed into the system controller (*DAQ*). The vector values are controlled by the robotic assessment handle (line 4). This tool can be regarded as a joystick, which is able to apply external forces or change the spring stiffness value in real-time, respectively.

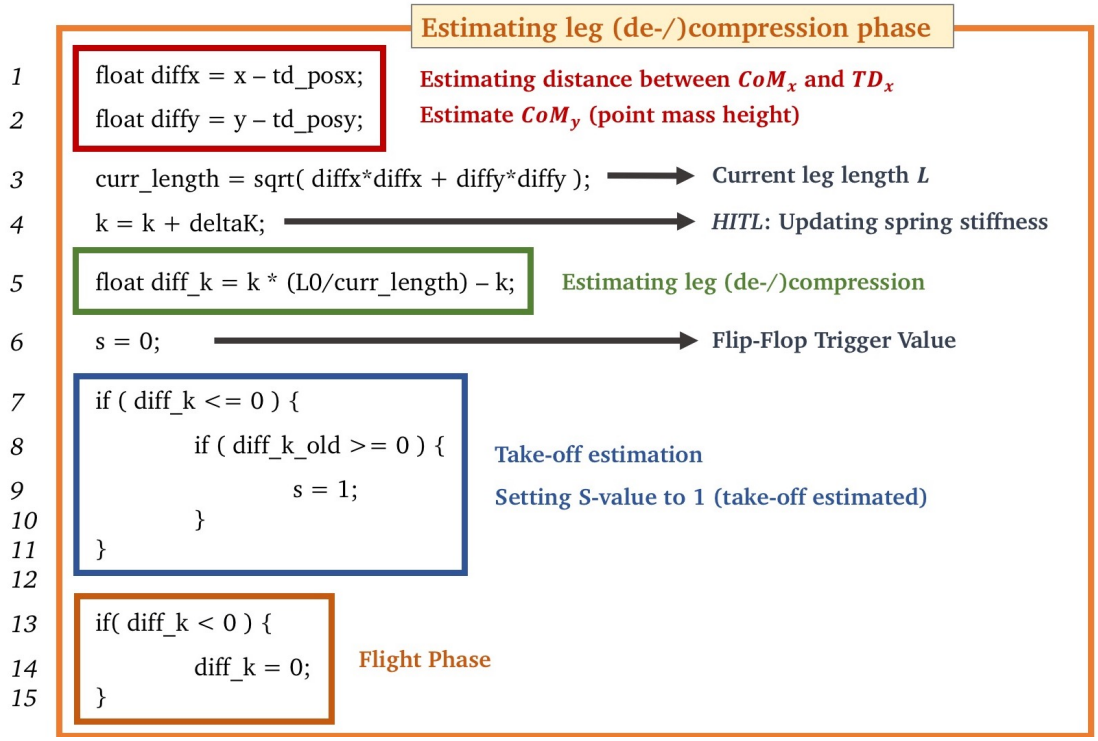

Figure 15: Estimating leg (de-/)compression phase

After the spring stiffness factor k gets updated, the de-/compression state of the virtual leg is determined. Further information and descriptions about running phases during a locomotion cycle is provided in Fig. 2.

Regarding line 5 (Fig. 15), the variable $diff\_k$ is declared and assigned to a certain value. If the variable $diff\_k$ becomes negative, the current leg length ($curr\_length$) is greater than $1\,m$ (= leg rest length $L0$). In line 6 of Fig. 15, the S-value of the S-R Flip-Flop module is set to 0. The S-value only turns to 1, if a take-off is estimated. Whether there is a take-off is checked in the following if-clause (lines 7-11). The previously declared and assigned $diff\_k$ variable (line 5) is compared to value 0. If the current variable value is equal or less than 0, and the value – set one iteration step before – was equal or greater than 0, the S-variable switches to 1. This happens once within each running cycle. The last three lines serve to check, whether the *SLIP* running model is in flight phase ($diff\_k < 0$). During this phase, the $diff\_k$ value is set to 0, because the current leg length L cannot exceed the leg rest length $L0$.

In order to calculate the resulting spring force ($f\_springx$, $f\_springy$) and the acceleration, the values $diff\_k$ (defined in line 5, Fig. 15) and $diffx/diffy$ (defined in line 1-2, Fig. 15) are multiplied (SIM2-3). The acceleration (ax, ay) of the point mass can be retrieved by dividing the spring force (x- and y-component) by the constant mass m, which is set to 80 kg. This value doesn't change during the simulation (SIM4-5).

$$float\ f\_springx = diff\_k * diffx; \quad \text{(SIM2)}$$

$$float\ f\_springy = diff\_k * diffy; \quad \text{(SIM3)}$$

$$float\ ax = f\_springx/m; \quad \text{(SIM4)}$$

$$float\ ay = f\_springy/m; \quad \text{(SIM5)}$$

At the end of each simulation step, the acceleration values (ax, ay) are fed as new input parameters to the next iteration. This process keeps on continuing until the simulation terminates. The simulation terminates either manually, or if the *SLIP* running model gets unstable and falls down to the ground line. More information about termination conditions can be found in chapter 4.4 (*Termination Conditions*).

5.8. S-R Flip-Flop Module

For triggering transitional events, such as the determination of a touchdown (of the *SLIP* running model) or take-off, respectively, a S-R Flip-Flop module has been implemented. Generally, S-R Flip-Flip modules are used in many sequential logic circuits. Two inter-connected logic gates form the basics of this circuit, whose output has two states. When the circuit is triggered into either one of these states by a suitable input pulse, it will save that state until it is changed by a further input pulse, or until power is removed.

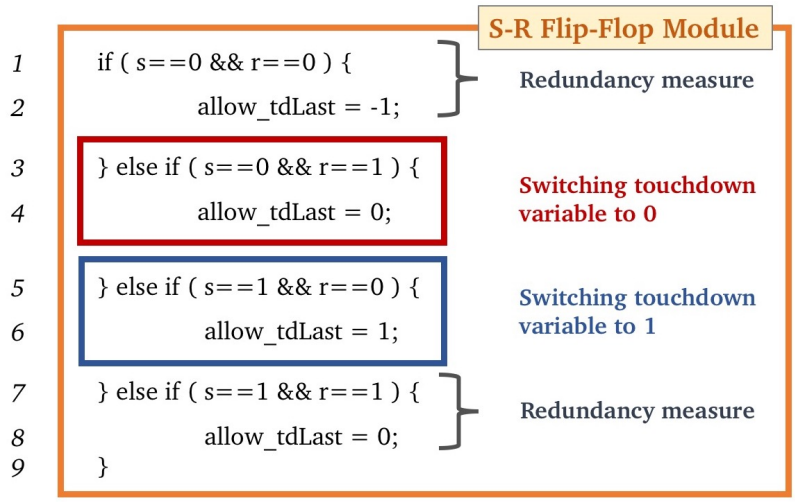

Figure 16: S-R Flip-Flop module for transitional event triggering

With respect to Fig. 16, the described *S-R* Flip-Flop module has been implemented by using a fourfold if-clause. First and last case are redundant and appear for the sake of completeness within the if-clause. The parameters S and R are trigger variables, one for the estimation of the take-off (S) and another for determination of the touchdown transitional event (R). Generally, before processing the *S-R* Flip-Flop module block, the variable $allow\_tdLast$ should be reset to value 0 ($allow\_tdLast = 0$). Consecutively, the main coding part is in lines 5-6 (Fig. 16), where a take-off has been estimated and the variable value of $allow\_tdLast$ switches from 0 to 1.

Since the Flip-Flop coding block is usually designed to save variable states, the variable value of $allow\_tdLast$ is fed as input parameter to the variable $allow\_td$ of the upcoming iteration step. In this case, the previously computed value (in one iteration step before) of $allow\_td$ is used. Logically, the *S-R* Flip-Flop module aims to switch values for the variable $allow\_td$ from 0 to 1 and vice versa. Setting $allow\_td = 1$ occurs one iteration

after a take-off has been determined by the *SLIP* running system. This value is kept at 1, until a touchdown is estimated. There is again a delay of 1 iteration, because the *S*-*R* Flip-Flop module block only computes previously calculated variables.

6. Results

In this chapter, the results of the simulation experiments are presented. The goal to be achieved within this work is to examine a simple *SLIP* model while running under unstable condition sets. Therefore, certain initial parameters have been changed at the beginning of a specified simulation set of trials, e.g. trials with a different set of initial horizontal velocity values (v_x) have been executed. Following values have been used for the simulation experiment: $v_x = 2.0\,m/s,\ v_x = 2.5\,m/s,\ v_x = 3.0\,m/s, v_x = 3.5\,m/s, v_x = 4.0\,m/s$. Since the haptic control design (chapter 3) within this work is based on the human-in-the-loop (*HITL*) approach, the subject is able to modify certain parameter values of the *SLIP* model in real-time. Beside the manipulation of the virtual stiffness factor k (initial value: 22,500 N/m), the *HITL* is able to apply an external force (F_ext) to the point mass in real-time (5.4 *Enabling Haptics*).

In chapter 6.1 (*General trials*), the simulation results are illustrated (Fig. 17, 21) in a x-y-diagramm showing the point mass trajectory over the achieved running distance. The sections 6.2-6.4 show simulation results of *SLIP* running under unstable initial conditions while using haptic device control.

6.1. General trials – No haptic input by the user

Within this section, experimental trials were performed, where no haptic input is introduced by the user, i.e. haptic device control is disabled during these experimental trials. There was neither a change in the stiffness, nor has an external force been added during stance phase to the point mass.

A distinction is made between two changing (initial) variables. As previously mentioned, the simulation has been performed by using different start values for the horizontal velocity v_x and the vertical *CoM* position y_0. Both scenarios, are explained in the next sections.

6.1.1. Changing initial horizontal velocity

The following Fig. 17 shows that the initial velocity v_x has a major impact on the behaviour of the *SLIP* model. Starting the simulation with an initial velocity of 5 m/s, the *SLIP* model immediately returns into a stable state after the first touchdown. The Apex height does not vary with regards to the steps made. By decreasing the initial velocity value v_x, it can be seen, that the system gets unstable and the Apex height increases significantly with every step made during the simulation, until the *SLIP* model falls down. The slower the running model starts; the less steps can be achieved. This is due to minor kinetic energy provided to the system.

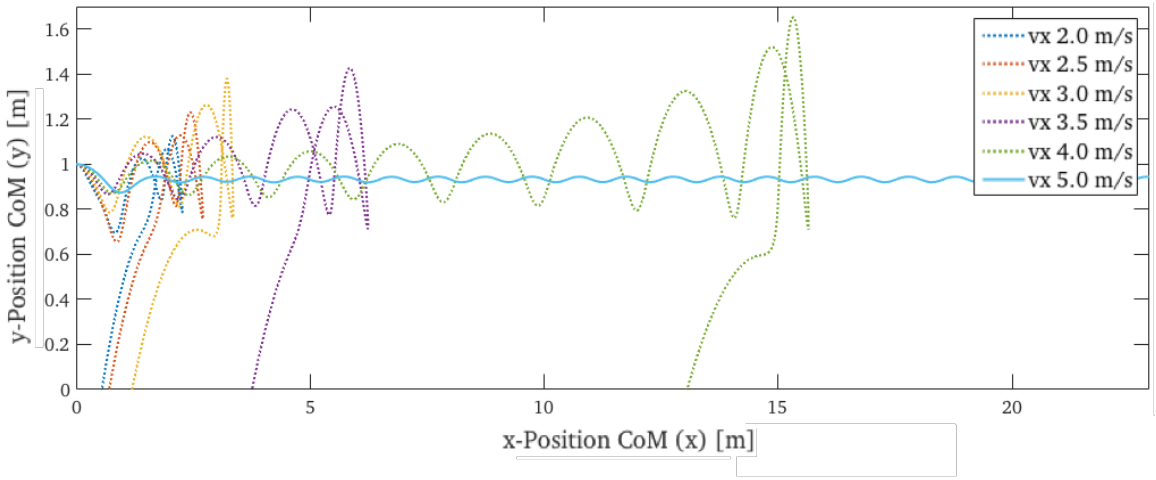

Figure 17: Changing initial velocity conditions without haptic input by the user

6.1.2. Changing initial height

Without haptic influence (i.e. adding an external force or changing the spring stiffness parameter) from the user's side, the *SLIP* model falls down for various initial height conditions. It should be noted that only the height condition has changed, all other start parameters are set to their respective default values. Taking the horizontal velocity as an example, the initial value is set to 5 m/s.

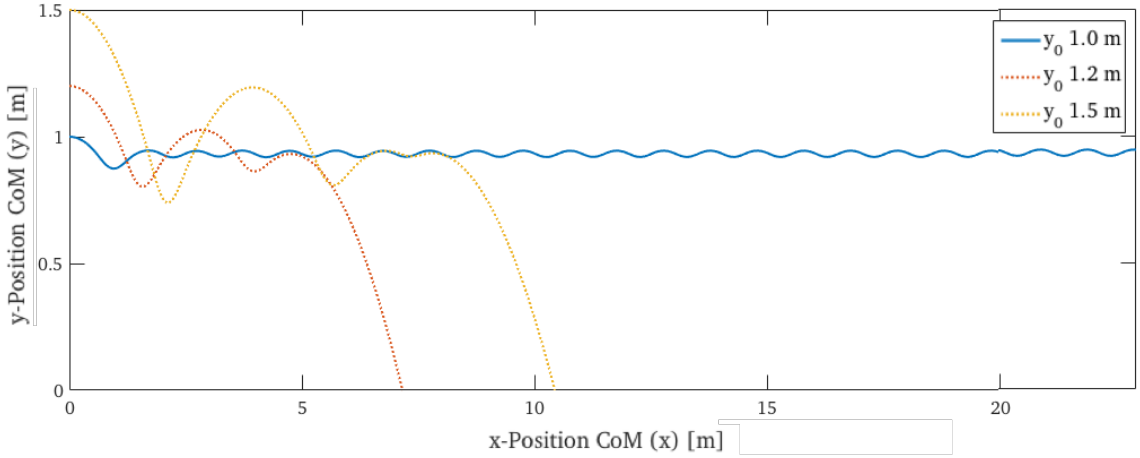

Figure 18: Changing initial height conditions without haptic input by the user

Regarding Fig. 18, it should be noted that the optimum height is 1 m. In this particular case, the foot of the *SLIP* model doesn't touch the ground yet, even if the y-position of the *CoM* is set to 1 m as well. This is due to the angle of attack, which is set to 70° by default. Furthermore, the impact of the first touchdown rises with increasing initial height conditions. The explanation is obvious, since during flight phase only gravitational force is acting on the point mass. Neglecting aerodynamic frictions, the vertical point mass velocity decreases continuously, until the first ground contact takes place, thus intensifying the impact.

From this point onwards, the current horizontal velocity, as well as the spring stiffness are decisive factors for the further curve trajectory of the *CoM*. More information on the importance and function of model parameters can be found in chapter 4.1 (*Main Variables*).

6.2. Trials with external force application

In the following, experimental results with an external force application to the point mass are shown. Based on the current angle α between leg axis and the ground line, the externally applied force is split into a horizontal and a vertical component. In total, 5 different constellations, with regards to changing velocity v_x, are examined. Since the *SLIP* model is very stable for an initial velocity of 5 m/s, this case is left out. It is more relevant to see, how the SLIP model behaves, when decreasing horizontal velocity at the beginning of the simulation.

Furthermore, it can be seen, that with an increasing number of attempts, the *SLIP* model improves, but it still falls down after a certain amount of steps (i.e. 6 max. steps). Regarding Fig. 19 (4^{th} trial, 5^{th} trial), the running model was able to achieve a distance of approximately 4.5 m within a time range of 3 s. Since this is the first series of trials (in total 5 trials), a major improvement with regards to the achieved running distance is noticed. All experimental trials are performed on a single subject. Therefore, it might be interesting to compare sensimotor learning results with more than one person.

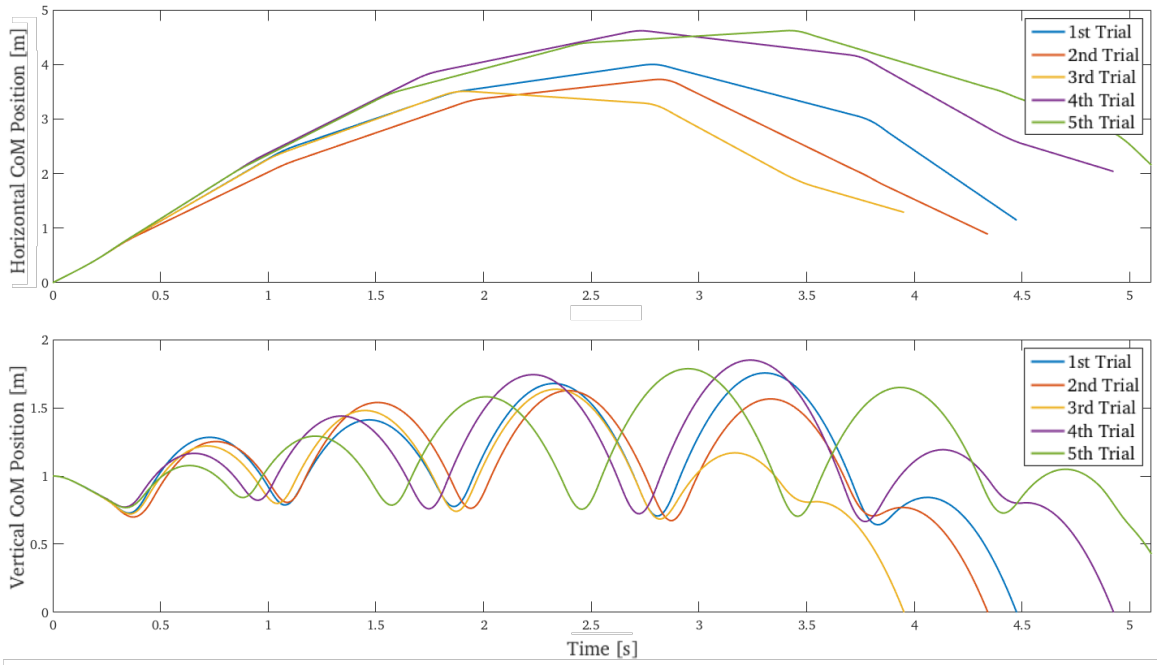

Figure 19: Haptic force application – 5 trials with $v_x = 2.0\ m/s$

By increasing initial velocity v_x to $2.5\,m/s$ (Fig. 20), the running distance is approximately twice as big as in the previous case ($v_x = 2.0\,m/s$). Also the amount of steps increased from 6 to 9 steps. Unfortunately, the system crashed during the 4th trial.

The *CoM* y-trajectory, as well as the change in Apex height is similar to the curve characteristics of the green trajectory (5th trial). Again the 3rd trial is the worst one regarding the achieved running distance. It should be noticed, that the concept of sensimotor learning is applying. Thus, the subject was able to respond more smoothly to the impacts and ground reaction forces occurring during stance phase.

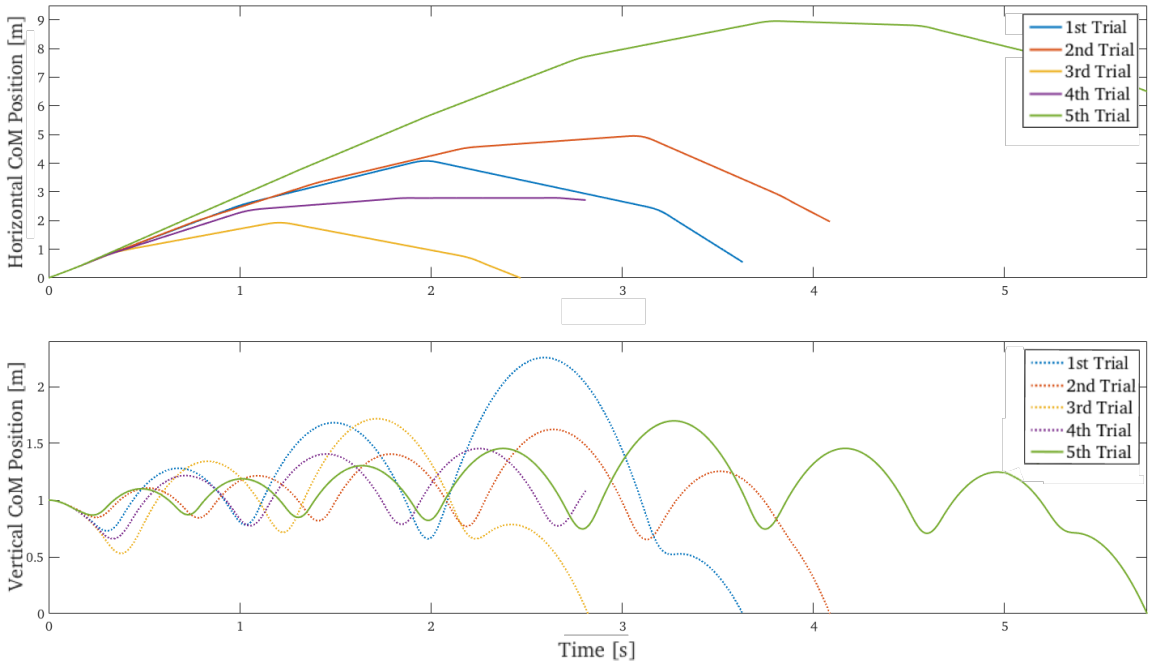

Figure 20: Haptic force application – 5 trials with $v_x = 2.5\,m/s$

Adding another $0.5\,m/s$ to the initial velocity v_x (Fig. 21), the systems seems to get stable. With regards to the upper subplot, it can be seen, that the *SLIP* runner is running forwards during the first 4 simulation seconds. Afterwards, the system remains stable, but the *SLIP* model is running backwards due to the rapidly decreasing angle of attack during the touchdown phase. The model doesn't fall down, because an external force is applied continuously from the touchdown to the take-off. Even if the angle of attack α_0 is set to 70° (4.1 *Main Variables*), the model isn't able to run forwards. 14 steps are counted in Fig. 21, but only the first 9 steps might be treated as valid values. In real world scenarios,

it is impossible to touch down with an angle of 70° (without kinetic energy in horizontal x-direction) and therefore to remain in the same position while jumping up and down. The 4th trial is comparable to the previous case ($v_x = 2.5\ m/s$). Differences are noticed in fluctuations of the Apex height change, which seems to be more stable for an initial velocity $v_x = 3.0\ m/s$.

Based on these two experimental cases, a hypothesis may be formulated stating that increasing Apex height changes might cause unstable point mass trajectories during running.

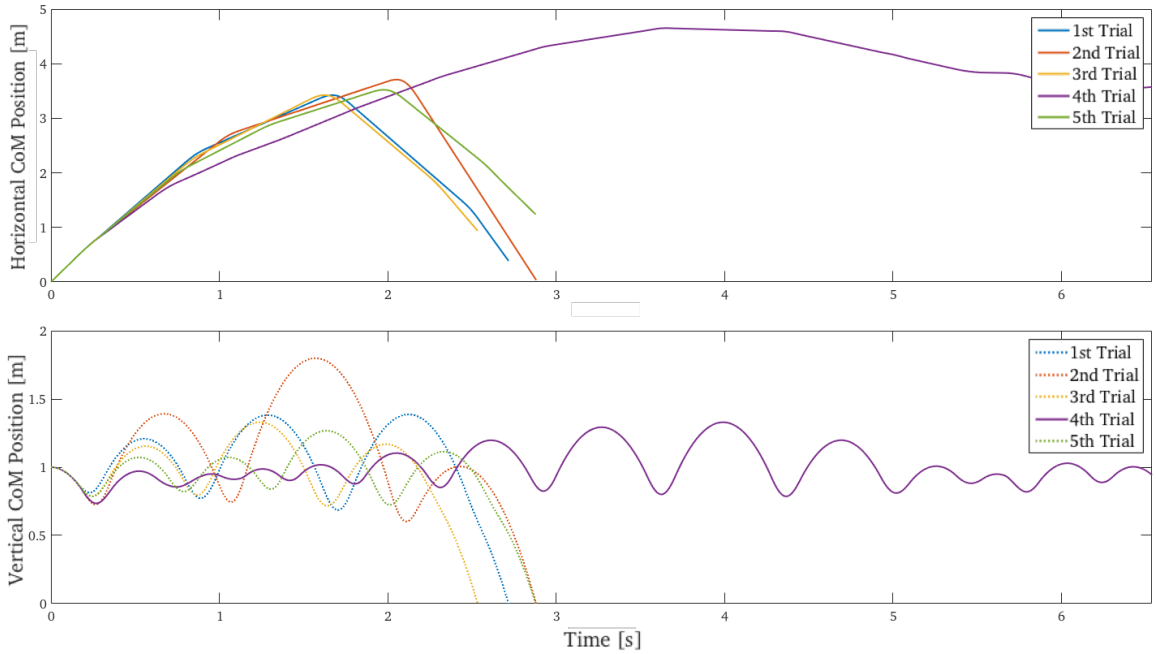

Figure 21: Haptic force application – 5 trials with $v_x = 3.0\ m/s$

The results for running with an initial horizontal velocity of $v_x = 4.0\ m/s$ seem to be more constant, than in previous trials. It should be noted, that the deviation between the smallest (3rd trial, yellow line) and the greatest (4th trial, purple line) running distance is approximately $2.5\ m$ (Fig. 22). Within this series of trials, the *SLIP* model reached more than $10\ m$ of running distance twice. Comparing this to previous results ($v_x = 2.0\ m/s$ and $v_x = 3.0\ m/s$), where the maximum running distance is about 4-5 m, a big difference is noted. The series with an initial velocity state of $v_x = 2.5\ m/s$ (Fig. 20), where a running distance of $9\ m$ has been reached, forms the only exception. This result can be

treated as an outlier, because within all other trials a significantly smaller running distance ($1 - 4\,m$) has been achieved.

The *SLIP* model shows a more stable locomotion pattern with an increased start velocity. If the initial velocity v_x gets too high (i.e. significantly higher than $5.0\,m/s$), the system returns back to an unstable state as well.

As part of this last series of trials ($v_x = 4.0\,m/s$), all 5 attempts show great results. According to the principles of trial and error, the user learned how to apply external forces with each running cycle. Of course, the good results are also due to the increased start velocity v_x. Nevertheless, the *SLIP* model still falls down after at least 6 steps. It would be interesting to see, how the sensimotor learning progress looks like after more than only 5 trials. Maybe a trained user is able to achieve a greater running distance or to stabilize the locomotion pattern, that no big Apex height changes are noticed. This and many more ideas can be read in chapter 8 (Future Work).

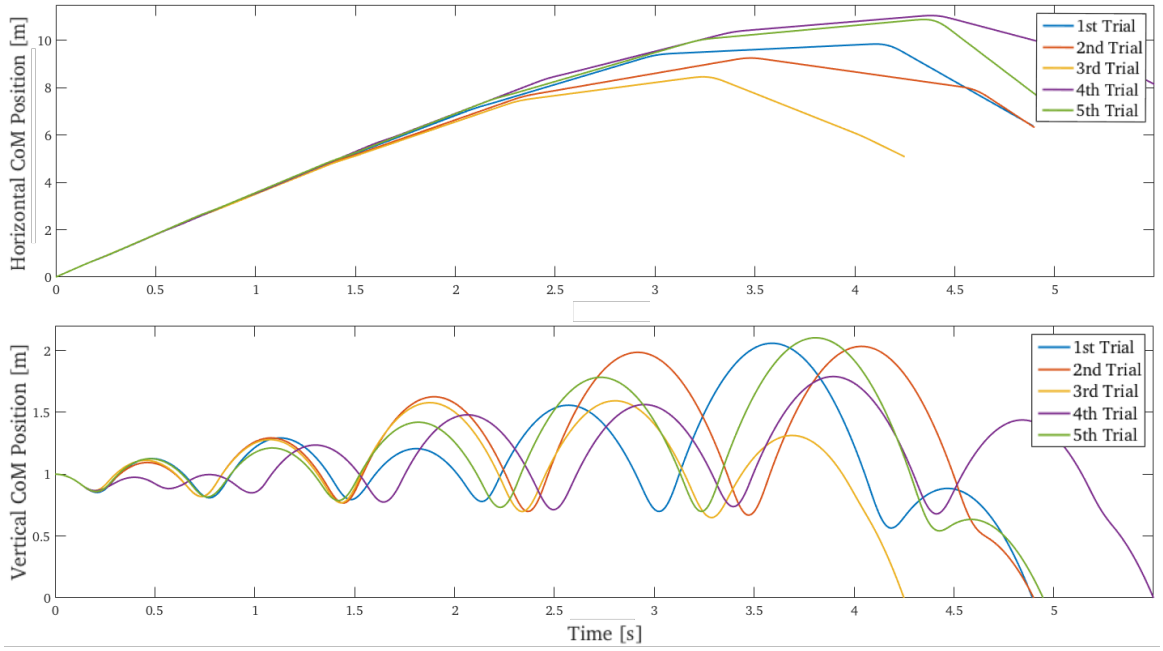

Figure 22: Haptic force application – 5 trials with $v_x = 4.0\,m/s$

6.3. Trials with spring stiffness manipulation

The following results show the *CoM* trajectory (related to the achieved running distance) with respect to a – in real-time – changing spring stiffness parameter during stance phase. This time, no external force is applied to the point mass; only internal forces act on the *CoM*, which are produced by a massless prismatic spring. In the first series of trials (with $v_x = 2.0\ m/s$; Fig. 23), a maximum running distance of 5 m is reached by the *SLIP* model. After the first attempts, the user roughly learned how to manipulate the spring stiffness parameter with respect to stabilizing the locomotion pattern. Hence, the last trial (number 5) has the longest simulation time. In this attempt the *SLIP* model didn't fall down. However, the 1st trial appears to be more successful, even though the running distance of approximately 4 m is 1 m less than in the 5th trial. This is due to the smooth *CoM* trajectory (blue line) in trial number 1, where no big Apex height changes occur, unlike in the *CoM* trajectory (green line) in trial number 5.

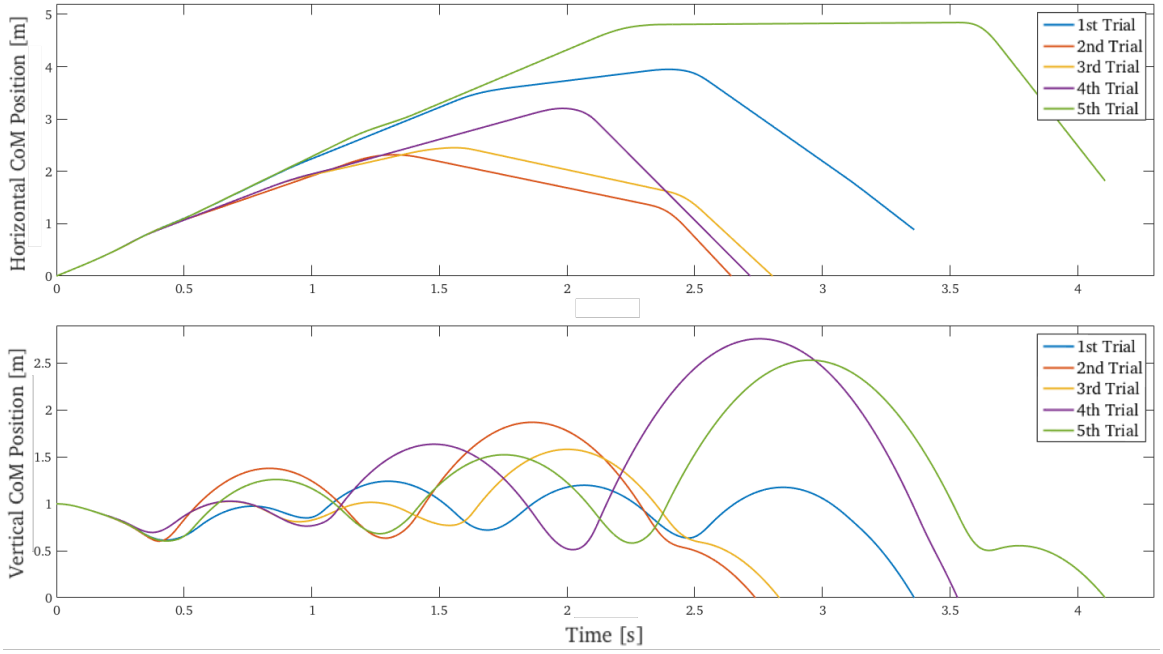

Figure 23: Haptic stiffness manipulation – 5 trials with $v_x = 2.0\ m/s$

Regarding Fig. 24, a difference is noted by increasing the initial horizontal velocity to $v_x = 2.5\ m/s$. The *SLIP* running model falls down immediately after the first step has been made. Thereafter, considering the last attempt (purple line), more than 30 m running

distance are achieved by the model. Comparing this results to the prior experimental series of trials ($v_x = 2.0\ m/s$), there are certain similarities. This can be seen as an effect of sensimotor learning. Comparing outcomes to the previous setup, where external forces have been applied to the point mass with a maximum running distance of roughly $10\ m$, results show a triplication of the distance accomplished by the *SLIP* model. Under previous initial conditions, where $v_x = 2.0\ m/s$, all trials terminate with a *SLIP* model jumping and falling backwards (see upper subplot). In this setup ($v_x = 2.5\ m/s$) – apart from trial number 2 – the *SLIP* runner falls down when running/jumping forwards.

Considering the purple line (4th trial) in Fig. 24, after approximately $4\ s$ of simulation time, a minimum change of the Apex is noticed. After $5.5\ s$, the *CoM* trajectory (lower subplot) of the *SLIP* runner shows growing fluctuations between Apex and touchdown position of the *CoM*. It is obvious, that the spring stiffness has changed drastically, due to the fluctuations.

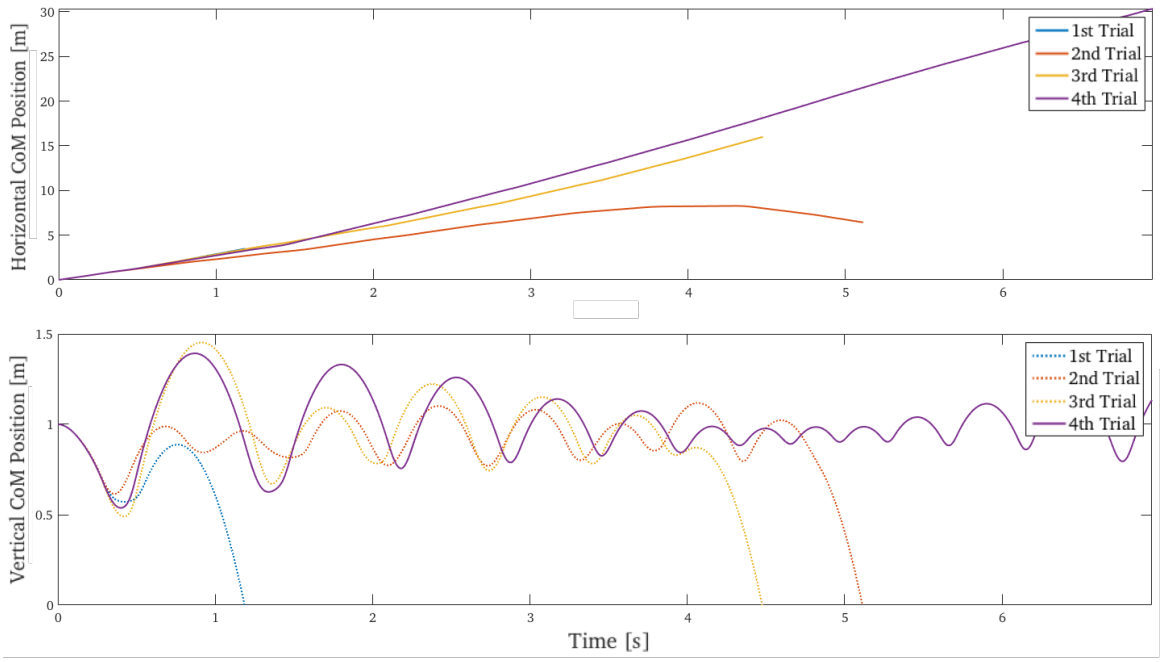

Figure 24: Haptic stiffness manipulation – 4 trials with $v_x = 2.5\ m/s$

According to Fig. 25, it can be assumed that the *SLIP* model is going to fall down within the next simulation seconds. After two unsuccessful trials, the following three attempts show roughly normal *CoM* curve trajectories, despite the big Apex height fluctuations. These fluctuations are apparently the reason for the future fall-down of the *SLIP* model.

Nevertheless, a running distance of over 20 m, has been achieved three times. It would be interesting to find out, how sensimotor training can affect the user customized input fed to the haptic device robot. Basically speaking, it is possible to achieve a totally stable running pattern with an initial horizontal velocity $v_x = 3.0\ m/s$.

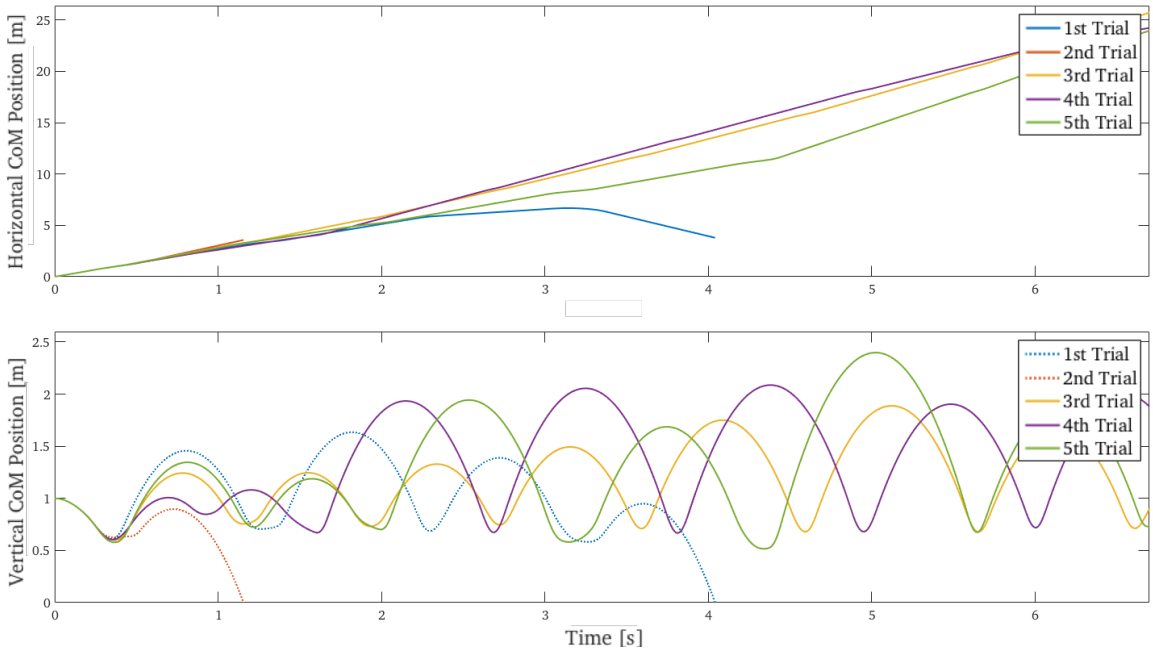

Figure 25: Haptic stiffness manipulation – 5 trials with $v_x = 3.0\ m/s$

According to the scenario with an initial horizontal velocity v_x of 4.0 m/s, within the first trial, a little more than 10 m distance has been reached. Afterwards the *SLIP* model falls back and the simulation terminates, if the *CoM* y-value is below 0. A stable locomotion is reached in the 4th trial. After increasing changes in Apex height (first 3 s, Fig. 26), the user figured out an appropriate strategy how to stabilize the running pattern.

A proper solution might be to decrease spring stiffness during the compression phase. Before take-off, the spring stiffness should be increased. There are many possible approaches, e.g. increasing the stiffness immediately before touchdown and reducing it while the spring length is minimized. This opens a new field of research considering the control of the real-time changing spring stiffness value during stance phase while running. Further interesting topics can be found out in chapter 8 (*Future Work*).

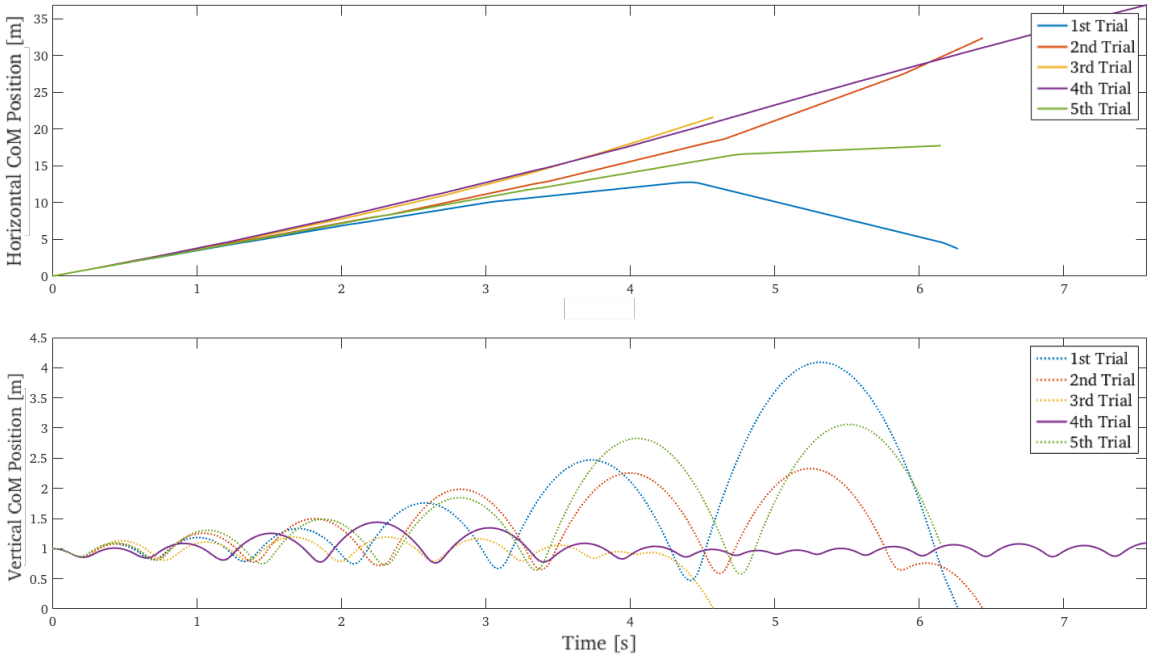

Figure 26: Haptic stiffness manipulation – 5 trials with $v_x = 4.0\ m/s$

6.4. Trials with spring stiffness manipulation and different height

In the following experimental trials with spring stiffness manipulation and different initial state height (y_0) of the point mass are examined and evaluated.

As it can be seen in Fig. 27, first two trials are almost stable and no big Apex height change is noticed. In each of the five trials, the *SLIP* model is running forwards (upper subplot) with an achieved running distance of almost 40 m, reached in the first two trials. Within the last three trials a maximum running distance of almost 15 m has been reached. Thus, while the first attempts more than 15 steps have been made in the first attempts, but afterwards the system falls down after approximately 3 steps, due to inappropriate haptic stiffness regulation at the beginning of the simulation (i.e. first touchdown).

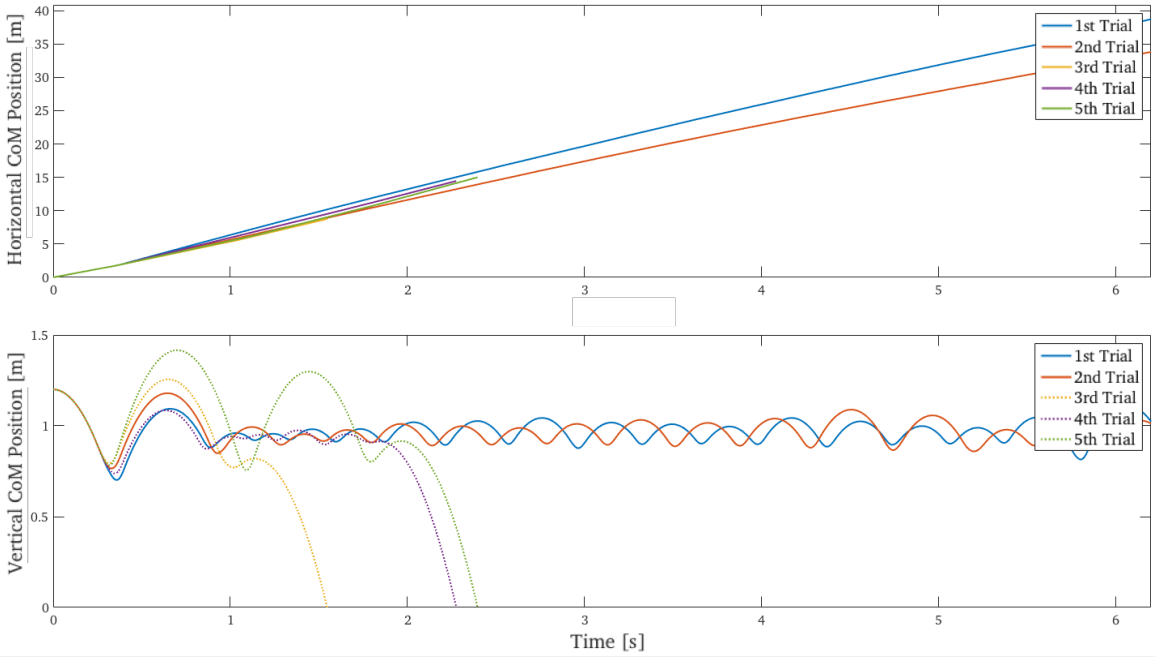

Figure 27: Haptic stiffness manipulation – 5 trials with *CoM* height $y_0 = 1.2\ m$

The same hypothesis applies, when using an initial *CoM* height of 1.5 m. The first impact within the 3rd trial is very smooth compared to the first attempt, where the *SLIP* model falls down after the third step. 10 steps are made in the following trial, then the model falls down as well. Before the fall-down (Fig. 28, red line), the curve characteristics between the red and the yellow line are very similar. In one scenario the *SLIP* running model achieves a stable state.

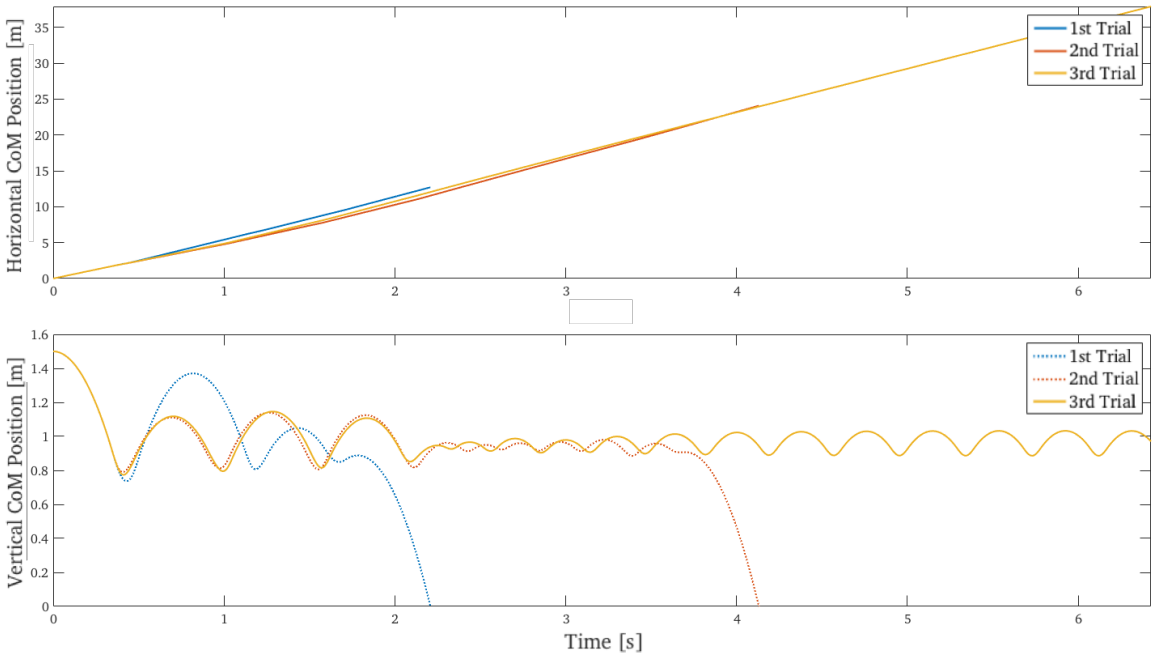

Figure 28: Haptic stiffness manipulation – 3 trials with *CoM* height $y_0 = 1.5\ m$

Since the main topic of this thesis is haptic device control with regards to velocity changes in a *SLIP* model, the various running scenarios with changing initial *CoM* height are only described marginally. The idea is to point out further investigation opportunities when modifying another initial state parameter (i.e. *CoM* height) within the *SLIP* model. More information on this topic can be found in the following section.

7. Interpretation & Discussion

The goal of this section is to provide the major insights gained from the experimental simulation (section 2.8) conducted within the scope of this work. It is natural, that user input commands may deviate each running cycle. This might lead to unstable running conditions, such as increasing Apex height changes. Human input commands can be trained based on sensimotor learning tasks, in order to feed the haptic device controller more precisely.

According to the results presented in the previous chapter 6 (Results), changing stiffness parameters has a more successful influence when it comes to achieving long running distances. Maximum running distance achieved with external force application was about $10\,m$, whereas stiffness manipulation scenarios triple this value, even for slow starts (i.e. $v_x = 2.5\,m/s$). Slow starts are directly related with velocity values under stable running conditions and are characterized by relatively small start velocities. In this particular case, dividing the initial horizontal velocity v_x by two, still leads to remarkably high running distances ($> 35\,m$) achieved by the user. By applying the human-in-the-loop approach, stable running with different initial parameters is possible, since the user is controlling the haptic device robot properly. Within the following trials, where spring stiffness is manipulated by the user ($v_x = 3.0\,m/s$ and $v_x = 4.0\,m/s$), results are more consistent and improving. Taking the setup with an initial horizontal velocity $v_x = 3.0\,m/s$ as an example, in three out of five trials a running distance of $25\,m$ has been reached. In these three trials, the *SLIP* model falls down while moving forwards, unlike in some scenarios where external force has been applied haptically.

When applying forces instead of manipulating the spring stiffness, the angle of attack ($\alpha_0 = 70°$) is the same at the beginning of each touchdown. After the first ground contact has been computed, the applied force is divided by trigonometric calculations. Instead of using a constant value for the angle of attack α_0, a possible extension to the running system might be to implement the flight dynamics of the *SLIP* model, where the leg orientation is taken into consideration as well (8 *Future Work*).

As investigated in the result section (6 *Results*), increasing Apex height changes lead to unstable running conditions. This makes sense, when using static parameters that don't change over time during the simulation process. It can be stated that increasing initial velocity minimizes *CoM* fluctuations. This applies until the optimal start velocity (approximately $5\,m/s$) has been reached. If this value is exceeded, further investigations need to be made in order to answer upcoming question concerning the *CoM* trajectory.

An interesting question to be answered within the scope of this work is, whether manipulating spring stiffness or applying an external force to the point mass during stance phase leads to a more stable running pattern. As shown above, changing the spring stiffness in real-time shows better results with respect to the achieved running distance. This is due to the fact, that changing spring stiffness simulates the biomuscular behaviour better with respect to muscle force production and compensation. Mechanically speaking, muscle force production refers to leg actuation while compensation refers to damping during stance phase. When actuating the virtual representation of the human leg, energy is introduced into the system and the prismatic spring leg gets stiffer. Based on these assumptions, inserting energy into the system during late stance may lead to greater Apex heights depending on the take-off angle.

A stable running pattern can be achieved under the systematic influence of parameter manipulations. Once the locomotion pattern of the *SLIP* model is regarded as stable, parameter changes may be repeated the same way during each running cycle. The first touchdown of the *SLIP* model poses a major challenge. Ideally, Apex heights changes differ less while the simulation is executed. An example is given in Fig. 24, where the purple line shows the greatest impact (i.e. ground reaction forces) while the first step is made. The *SLIP* model seems to compensate the impact of the first touchdown successfully, considering *CoM* trajectory in y-direction in lower subplot, at the beginning of the simulation, afterwards it seems, that the running model returns back to unstable state.

Due to *LabView*-intern operational issues with formula nodes (script nodes for mathematical expressions and calculations) used in the *SLIP* simulation developed within the scope of this work, certain changes needed to be made. The sampling rate was set to $1\,ms$, which caused the system to crash at the beginning of the simulation. By setting the sampling rate to $2\,ms$, the system simulation was running fine for a sufficient amount of time. Luckily, the trajectory of the *SLIP* model did not vary for the same start parameter set, after this change has been made. Changing the sampling rate may affect the system, if the simulation time increases drastically.

Beside the upper mentioned case, no further issues occurred during the experiments.

8. Future Work

This chapter presents new topics related to the research field of haptics and general biomechanics and deriving from the experiments the experiments described in this work, which are designed to answer more questions than approached in this thesis. Since this work is dealing with interactions between a single human and a machine, it might be interesting to include a second haptic device controller to the system environment. Melendez-Calderon (2011) already introduced a novel, versatile dual wrist interface at the International Conference on Intelligent Robots and Systems (*IROS*), in order to investigate the control of human-human or bimanual interaction. His goal was to investigate the mechanisms of human-human interaction in collaborative and competitive tasks. Knowledge gained from the experiments with the dual-wrist haptic device controller, can then be used for consecutive experiments. As this Thesis is only covering the simulation for one-legged running, the next step would be to extend this model into a two-legged running model.

It would be interesting as well to see the model walking, instead of running. In this particular case of human locomotion, it is necessary to consider the second leg for the calculation of the movement trajectories. While walking both feet touch the ground at some point in time. Therefore, it is mandatory to work with a two-legged model.

Another interesting perspective would be to exchange the spring for a Hill-type muscle model, in order to see, how the different components affect the movement dynamics of running, respectively walking.

During each experimental trial, the start configuration remains the same, except for two initial parameters: The horizontal velocity v_x and the initial height y_0 of the *CoM*. Within the scope of this thesis, the variety of different parameter sets as initial values is limited. The main goal of this work was to implement a *SLIP* model and to couple it with a one-wrist haptic device controller. Subsequent works may use this *SLIP* model implementation with different parameter sets. Therefore, it may be interesting to discover new initial state values, which are also suitable for stable running, e.g. the initial vertical velocity, which has been set to 0. Setting a different value could give some indication of what would happen, if the *SLIP* model is already falling. Will this model fall down immediately, or is it possible by the means of haptic interaction to increase the spring stiffness within the decompression phase during stance? Are there more parameters, than adjusting the stiffness of the spring or adding an external force to the equation of motion, which may be modified in real-time? All these questions can be answered by setting up an

experimental design with a wide spread range of different parameter values. Exploring different initial state variables, might bring new insights into the theory of stable running. While manipulating variables in real-time, several trials should be executed with different start parameter sets. Initial state parameters should change within a rational range before each trial block, e.g. horizontal velocity $v_x = 4\,m/s$ may be combined with an untypical initial *CoM* height of $y_0 = 1.2\,m$. It would be interesting as well to manipulate various parameters simultaneously. Therefore, at least a two-dimensional haptic hardware controller is needed.

Furthermore, the *SLIP* model used within this work takes a fixed angle of attack (α_0) as a touchdown parameter during each ground contact phase. A possible model extension might be to develop a dynamically adaptable angle of attack, which is taking the leg orientation during the flight phase into consideration as well.

An interesting future work is to develop a multi-segmented *SLIP* model with at least a knee joint. Another important joint, that can be introduced to the system is the ankle joint. Simulations based on correct algorithms and calculations, would be able to represent the locomotion dynamics of the human lower limb more precisely. Therefore, a new experimental plan should be designed.

Eventually, the spacing between the velocity values can be reduced as well. Within the scope of this work, relatively big gaps between the velocity values during each setup have been used. A proper solution might be to reduce the incrementation interval to $0.1\,m/s$. Especially, when it comes to the examination of sensimotor learning tasks, it might be interesting to see, which initial state values lead to significantly better results. In both setup trials with an initial velocity of $2.5\,m/s$, where external forces have been applied, as well as the spring stiffness has been manipulated, the results deviate remarkably from the previous setups (i.e. $v_x = 2.0\,m/s$). This leads to the question, if there is a certain velocity value between these two initial state conditions, which facilitates the user to achieve a greater running distance. As mentioned before, it might be due to sensimotor learning skills as well. When running with an initial speed of $v_x = 2.0\,m/s$, the *SLIP* model falls down predominantly, because the user has a lack of experience manipulating the parameters in real-time. After the first setup is done, a certain controlling pattern is recorded by the user, which enables him to achieve more steps during a trial.

This work provides a general overview of the possibilities using a haptic device robot, and demonstrates the coupling with a biomechanical *SLIP* model. Additionally, the point mass trajectories could be evaluated in more detail with respect to topics such as sensimotor learning and training.

In order to treat haptic real-time manipulation as a sensimotor learning task, a single subject is not sufficient for the development of a hypothesis. Based on the human-in-the-loop (*HITL*) mechanism, which applies to the whole system setup, subjects generally improve the ability to command the haptic device robot with respect to achieve stable locomotion. Hence, a future work might setup an experimental design with multiple subjects (e.g. 8-10 subjects), who are trying to manipulate certain *SLIP* parameters during stance phase. After performing all experiments, it would be interesting to find out, if the subjects could learn to control the *SLIP* runner under unstable conditions.

References

Alexander, R. (1976). *Mechanics of bipedal locomotion. Perspectives in experimental biology* (ed. P. S. Davies), pp. 493–504. Oxford, UK: Pergamon Press.

Arslan, O. (2008). *Spring Loaded Inverted Pendulum (SLIP). Approximate Stance Map For Nonsymmetric Motions and Variable Stiffness*. Bilkent University, Ankara: Electrical & Electronics Engineering Department.

Blickhan, R. (1989). The spring-mass model for running and hopping. *Journal of Biomechanics*, 22 (11-12), 1217–1227.

B.A.L.A.N.C.E (2013). *Preliminary System Requirements and Evaluation Metrics Update M12*. A partially funded project by the Seventh Framework Programme (FP7) of the European Commision (FP7-ICT-2011.2.1) under Grant Agreement No. 601003. Last access on 19.11.2014 under http://www.balance-fp7.eu/.

Carignan, C. R. & Cleary, K. R. (2000). Closed-Loop Force Control for Haptic Simulation of Virtual Environments. *Haptics-e*, 1 (2), 1-14.

Farkhatdinov, I., Garnier, A. & Burdet, E. (2015a). Development and Evaluation of a Portable MR Compatible Haptic Interface for Human Motor Control. *IEEE Worldhaptics 2015*, Jun 2015, Chicago: United States.

Farkhatdinov, I. & Burdet, E. (2015b). *Haptic interaction models for learning stable locomotion.* London: Faculty of Engineering.

Fisch, A. J., Mavroidis, C., Bar-Cohen, Y. & Melli-Huber, J. (2007). *Development and testing of haptic interfaces using electro-rheological fluids. Chapter 4: Haptic Devices for Virtual Reality, Telepresence and Human-Assistive Robotics*. A dissertation submitted to the Graduate School. Boston: Department of Mechanical and Industrial Engineering, Northeastern University.

Geyer, H. (2005). *Simple models of legged locomotion based on compliant limb behaviour*. PhD Thesis. Jena: Fakultät für Sozial- und Verhaltenswissenschaften.

Hutter, C., Remy, D., Höpflinger, M. A. & Siegwart, R. (2010). SLIP Running with an Articulated Robotic Leg. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4934-4939. Zürich: ETH-Zürich.

Imperial College (2017). *Portable Hi5*. Unpublished specification paper on the haptic device interface developed by the Biomedical Engineering Group. London: Royal School of Mines.

Krishnapillai, D. (2015). *An investigation into the effects of haptic feedback on control of dynamic systems*. Contribution by the Department of Bioengineering to the MEng Individual Project. London: Faculty of Engineering, Imperial College London.

MacLean, K. E. (2008). Haptic Interaction Design for Everyday Interfaces. Chapter 5. *Human Factors and Ergonomics Society*, 5, 149–194.

Ma, L. (-). *Haptic Interfaces*. Presentation Slides of the IRCCYN. Nantes: EC.

McMahon, T. A., & Cheng, G. C. (1990). The mechanics of running: How does stiffness couple with speed? *Journal of Biomechanics*, 23 (suppl. 1), 65–78.

Melendez-Calderon, A., Bagutti, L., Pedrono, B. & Burdet, E. (2011). *Hi5: a versatile dual-wrist device to study human-human interaction and bimanual control*. San Francisco, California, USA: IEEE/RSJ International Conference on Intelligent Robots and Systems.

Moussette, C. (2012). *Simple Haptics. Sketching Perspectives for the Design of Haptic Interactions.* Thesis for the degree of Doctor of Philosophy in Industrial Design. Umeå, Sweden: Umeå Institute of Design, Faculty of Science and Technology.

Rossi, C. (2015). *How do humans learn stable locomotion?* Bachelor Thesis at the Department of Bioengineering (Imperial College London). London: Faculty of Engineering.

Seyfarth, A., Geyer, H., Günther, M. & Blickhan, R. (2002). A movement criterion for running. *Journal of Biomechanics*, 35, 649–655.

Schmuckler, M. A. (1990). Issues in the development of postural control. In H. Bloch and B. I. Bertenthal (Eds.), *Sensory-motor organizations and development in infancy and early childhood* (pp. 231-236). Doordrecht: Kluwer Academic Publishers.

Sharbafi, M. & Seyfarth, A. (2015). *FMCH: a new model for human-like postural control in walking*. Hamburg, Germany: International Conference On Intelligent Robots and Systems.

Tongen, A. & Wunderlich, R. E. (2010). *Biomechanics of Running and Walking*. Mathematics and Sports Theme Article. Last access on February, 3rd 2017 under http://www.mathaware.org/mam/2010/essays/TongenWunderlichRunWalk.pdf.

Wilhelm, E., Mace, M., Takagi, A., Farkhatdinov, I., Guy, S. & Burdet, E. (2016). *Investigating Tactile Sensation in the Hand Using a Robot-Based Tactile Assessment Tool*. London: Department of Bioengineering, Imperial College.

Wolpert, D. M., Diedrichsen, J., Flanagan, R. (2011). Principles of sensorimotor learning. *Nature Reviews*, 12, 739–751.