.24445

CONTROL OF SOFT ROBOTIC KNEE BRACE TREATING ACL LESION USING ITERATIVE LEARNING

J. (Jorn) Jansen

MSC ASSIGNMENT

Committee:
prof. dr. ir. S. Stramigioli
dr. ir. H. Naghibi Beidokhti
dr. ir. M. Abayazid
Y.X. Mak, MSc
dr. J. Reenalda

May, 2021

034RaM2021
Robotics and Mechatronics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

Anterior Cruciate Ligament (ACL) is one of the stabilizing ligaments present in the knee joint. An ACL rupture is the most common knee injury, dramatically altering the biomechanics of the knee joint. Most important decreasing the knee stability. Improper treatment has serious complications, such as osteoarthritis.

This study focuses on designing a control system for a pneumatically actuated robotic knee brace. The goal is to lift limitations, such as partial motion impairment and passive stiffness, of a traditional passive brace while returning healthy knee stability tailored to the specific patient.

At first the focus application is human gait (walking). The design is focused on exploiting the iterative behaviour of this activity and learning from past iterations to improve the control signal.

Due to the COVID-19 pandemic physical lab access was prohibited, as a consequence a specific knee model in Simulink is made to allow validation of the control design virtually.

Preface

Before you lies the research thesis "Control of Soft Robotic Knee Brace Treating ACL Lesion Using Iterative Learning". It has been written to fulfil the graduation requirements of the Electrical Engineering (Robotics and Mechatronics) Master Program at the University of Twente. The whole research was performed during the COVID-19 pandemic.

I would like to thank prof. dr. ir. Stefano Stramigioli for being the chair of this research thesis. I would like to thank dr. ir. Hamid Naghibi Beidokhti for supervising and guiding me on a daily basis and providing me with the basis of this research. I would also want to thank dr. ir. Momen Abayazid and Yoeko Mak, MSc. for providing feedback on the research and being part of the committee. Dr. Jasper Reenalda I would like to thank for being the external member of my committee.

Furthermore, I would like to thank my friends and family for supporting me throughout this process.

Jorn Jansen
Enschede, Wednesday 26th May, 2021

Contents

List of Abbreviations

ACL	Anterior Cruciate Ligament
ACLD	Anterior Cruciate Ligament Deficient
ACLR	Anterior Cruciate Ligament Reconstruction
AP	Anterior-Posterior (translation)[1]
IE	Internal-External (rotation)[1]
PAM	Pneumatic Artificial Muscle
SI	Superior-Inferior (translation)[1]
VV	Varus-Valgus (rotation)[1]

[1] Translation and rotation of the tibia with respect to the femur

1 Introduction

1.1 Overview

Anterior cruciate ligament (ACL) is one of the stabilizing ligaments present in the knee joint. When looking at knee injuries, an ACL lesion is the most common (Majewski et al., 2006). The effects of an ACL lesion can significantly alter the biomechanics of the knee joint through decreased stability, lowered proprioception, and increased laxity (looseness) of the joint (Lu et al., 2006; Mannel et al., 2004; Dhillon et al., 2011). Altered biomechanics of the knee joint with instability as a result often causes damage to meniscal and surrounding cartilage, which could later lead to osteoarthritis (OA) (Potter et al., 2011; Neuman et al., 2010; Mayr et al., 2010; Naghibi Beidokhti et al., 2020).

One of the current treatments for severe ACL lesion is ACL reconstruction, sometimes abbreviated as ACLR, by means of surgery. This surgery involves removing the torn ACL and replacing it with either a donor ligament (allograft) or with a tendon graft from another part of the knee (autograft) (Mannel et al., 2004; Macaulay et al., 2011). A traditional passive brace will aid in return stability post-surgery. For patients with minor stability loss and/or no to moderate sports ambitions might opt for a non-surgical approach by means of physiotherapy. This is referred to as an ACL deficient case, or abbreviated as ACLD.

Both ACL deficient and ACL reconstructed patients are encouraged to use a normal gait pattern for a faster return to overall leg strength (Lu et al., 2006). While the use of a traditional knee brace is encouraged, braces that cause limiting motion and hard limits at the extreme angles can result in non-natural gait pattern and might cause pain for the patient.

A robotic brace should be able to deliver said stability, increased proprioception, and decrease in laxity while not limiting normal gait and not causing any discomfort or pain to the patient or user. The approach to achieve these goals is by reducing the laxity caused by an ACL lesion in the tibial anterior-posterior (AP) translation and tibial internal-external (IE) rotation (Naghibi Beidokhti et al., 2020) while not limiting normal gait motion and improving proprioception by using soft robotic actuators embedded onto the brace. The soft robotic brace should also be able to achieve this, in contrast to a traditional passive brace, in a scenario where there is a transition from non-weightbearing to weightbearing, such as during heel-strike.

In case of an ACL deficient scenario, the brace will be used to mainly deliver stability to the patient by reducing the increased laxity while, as mentioned above, not limit the motion of the knee joint and cause any discomfort. For the ACL reconstructed case, the robotic knee brace should improve over a traditional knee brace in terms of protection of the graft. This should result in an increase in healing rate, decrease the chances of re-tearing the reconstructed ligament, and lower the damages to meniscal and surrounding cartilage. Furthermore, in both cases the soft robotic brace should be adaptable to deliver tailored care specific to that patient.

The motion of the knee is delivered mainly by the patient's muscles, the proposed research will be by adding force to the robotic knee brace to reduce the increase in tibial AP translation laxity and tibial IE rotation laxity caused by ACL lesion, thus increasing joint stability, increasing proprioception, and decreasing knee laxity. These forces will complement the patient's normal gait, not limiting the patient's motion, or causing discomfort or pain.

This research will be based on previous research performed on this specific soft robotic knee brace by Khambati (2019) and Ganesh (2020a). They performed research into the physical brace. The actuators used on the soft robotic brace is further researched by Ganesh (2020b). Knee laxity data obtained through research by Naghibi Beidokhti et al. (2017a) and Naghibi Beidokhti et al. (2020) will be used to create models of different cases: ACL intact and ACL deficient.

No specific control application has yet been developed for controlling the brace of Khambati (2019) and Ganesh (2020a).

1.2 Objective

The research question, and objective, of this thesis is how to recover healthy knee motion in an ACL deficient knee during daily activities by controlling pneumatic artificial muscles (PAMs) on a soft robotic knee brace. The aim of this thesis is to develop the control system that will control the PAMs which are on the knee brace. This control system must be able to aid the patient his/her knee during the daily activity of gait (walking). The robotic knee brace will aid the patient by applying a force in the AP direction, or by applying a moment in the IE direction, to correct the knee laxity caused by ACL deficiency. The brace could also be used to constrain the knee motion if necessary.

The healthy motion must be achieved while not obstructing any preserved healthy motion by the knee itself and only recovering the missing motion from the patient. Next to the main goal the transition of non-weightbearing to weightbearing can be examined while using the soft robotic brace, and if so be improved. The latter is considered to be a secondary goal of this research.

The brace should be worn during gait and return healthy motion during this activity. The clinician will be able to specify an AP force and IE torque profile. Based on this profile an AP and IE motion profile is computed and supplied as reference to the control system to serve as motion control. For now the focus is on gait, in the future other activities can be evaluated.

The way to validate if the goal is achieved is to compare the AP and IE motion of an ACL intact knee, ACL deficient knee, and ACL deficient knee with the soft robotic brace. All with the exact same external disturbances (forces as a result of the gait motion). The motion of an ACL deficient knee with soft robotic knee brace should be moved from the ACL deficient knee motion towards the ACL intact knee motion. This way it can be concluded that the soft robotic knee brace reduces the additional laxity caused by ACL deficiency.

1.3 Approach

The robotic knee brace used in this research will be actuated using PAMs which is developed by Khambati (2019); Ganesh (2020a). The focus here will be designing and implementing a control system to control the PAMs to add forces in the AP direction and moments to the rotation of the tibia to reduce laxity caused by ACL deficiency. These pneumatic actuators have been chosen because they are compliant and therefore provide safe actuation around the human body.

Due to the COVID-19 pandemic lab experiments are not possible. Because of this reason first a model will be made that reflects the knee joint sufficiently on which the control system can be evaluated. First, the focus will be on rehabilitation support for ACL deficient patients by reducing the increase in laxity, after the brace's (software) control can be customized and optimized for post-surgery usage for ACL reconstructed patients to protect the graft as it heals. Assumed in this case is that the physical brace is the same, but by adapting the control the brace must be able to suit to ACL deficient and ACL reconstructed patients.

Figure 1.1 shows a block diagram of the overview of the approach. This includes the knee model and soft robotic brace controller by the controller.

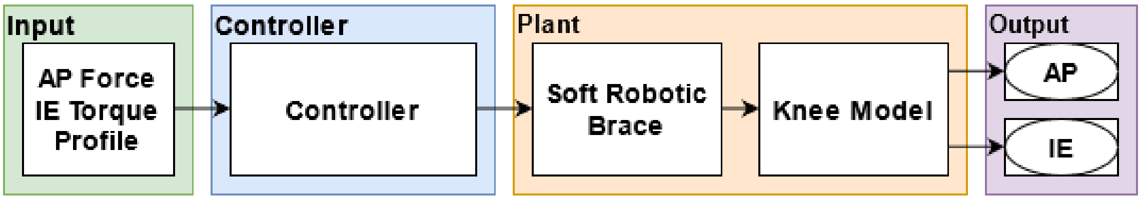

Figure 1.1: Block diagram of the approach

1.4 Outline

This research report starts with giving background information about the knee joint, the ACL and ACL lesion in Chapter 2. This chapter also contains an overview of the knee brace hardware. This includes the properties and design of the physical brace prototype together with the PAMs attached to the brace. Lastly, this chapter includes previous research on which this research is built upon.

Chapter 3 gives an overview of the simulation environment in which the control system will be simulated and tested. Due to the COVID-19 pandemic the physical tests were not possible. In this chapter a knee with attached pneumatic robotic brace has been simulated in a way to later verify the control. This assembly is further referred to as the plant. Besides a detailed overview of the plant, this chapter also discusses the plant verification to real-life data, its region of operation, and future improvements to the model.

Chapter 4 describes the controller in-depth. It highlights which control strategies are considered. The chapter gives a detailed overview on how the chosen control strategy is implemented and why such implementation is chosen.

In Chapter 5 results of the individual control system components and complete system are presented and discussed. A conclusion is drawn from these results and the goals from Section 1.2 are compared to the results in Chapter 6. Further research and the limitations of this research are discussed in Chapter 7.

2 Background

2.1 The Knee Joint

The human knee joint is one of the largest and most complex joint of the human body.[1] The knee joint connects the femur (thighbone) to the tibia (shinbone). Together with the patella (kneecap) and fibula it forms the knee joint.

Besides the bones, the knee joint consists of cartilage, tendons, and ligaments. One type of cartilage, articular cartilage, serves the purpose of smoothing the bone ends, allowing smooth motion. Another type of cartilage, the menisci, serves as load distribution and shock absorber (Fox et al., 2011).

Tendons, which connect bone to muscle, allow muscles to move bones. Ligaments connect bone to other bone. Ligaments have a restraining function, which helps the knee keep stability. Figure 2.1 shows an anatomic representation of the knee joint, including the different cartilage, tendons, and ligaments (Hoffman, 2019; OrthoInfo, 2014).

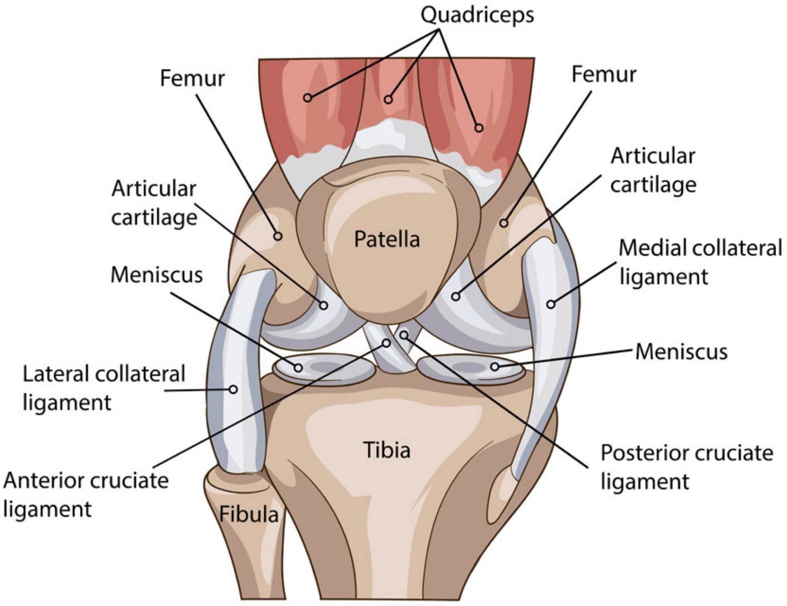
Quadriceps
Femur
Femur
Articular cartilage
Articular cartilage
Patella
Medial collateral ligament
Meniscus
Meniscus
Lateral collateral ligament
Tibia
Posterior cruciate ligament
Anterior cruciate ligament
Fibula

Figure 2.1: Frontal view of knee anatomy (Wilson, 2018)

2.1.1 Ligaments

As mentioned above, ligaments provide stability to the knee joint. This is called *passive* stability as mentioned by Masouros et al. (2010). The ligaments inside the knee joint can be divided into two groups: *Collateral ligaments* and *Cruciate ligaments.*

Collateral ligaments are found on the medial and lateral[2] sides of the knee joint. These two ligaments, seen in Figure 2.1, are responsible for the stabilization of side-to-side motion. The

[1] For this research the knee joint is described in such way that it is relevant for this specific research. The author is aware that the knee joint is more complex than described in this chapter.

[2] Medial meaning towards the middle of the body. Lateral meaning towards the side of the body.

cruciate ligaments are found inside the knee joint, also seen in Figure 2.1, in which they from a cross shape together. These two ligaments are the primary restraint of tibia translation in the anterior and posterior direction. In other words they prevent the tibia from sliding forwards and backwards on the femur, more specifically (Hoffman, 2019; Masouros et al., 2010):

- The anterior cruciate ligament, further referred to as ACL, prevents the the tibia from sliding anterior (forwards) on the femur. It furthermore acts as a restraint to internal tibial rotations at full extension and controls the tibial rotation during the last part of extension (screw-home motion).

- Respectively, the posterior cruciate ligament (PCL) prevents the tibia from sliding posterior (backwards) on the femur. It also has a restraining function for external tibial rotation.

2.1.2 Muscles

The quadricep (flexor) and hamstring (extensor) are the two major muscles groups involved in the knee joint. Flexor means that the muscle allows flexion of the joint, extensor on the other hand allows for extension of the joint. Together they form an agonist-antagonist pair and allow for flexion and extension of the knee joint.

The quadriceps and hamstrings are both connected to the bone with tendons, together they serve to move the tibial bone, but also keep the knee stable. The use of muscles to keep the knee joint stable is called *active* stability (Masouros et al., 2010). Both muscle groups and tendons can be seen in Figure 2.2.

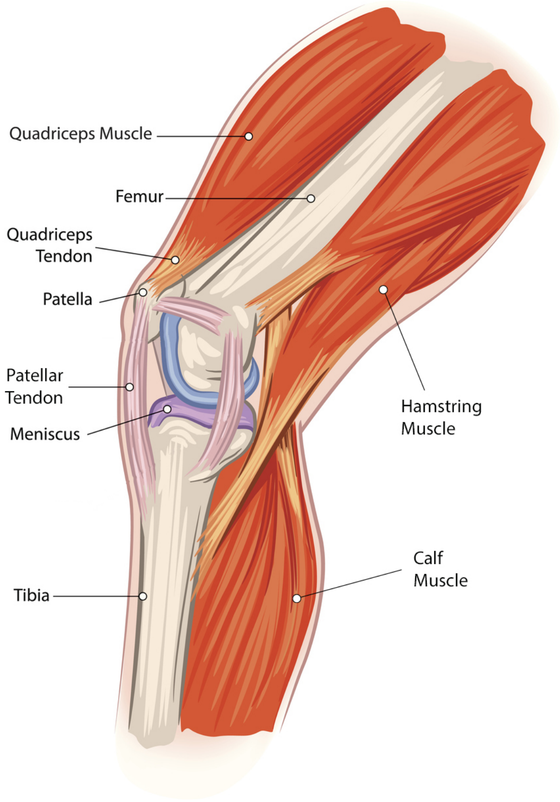

Figure 2.2: Isometric view of knee anatomy (Joint Health Matters, 2017)

2.1.3 Biomechanics

The main biomechanical role of the knee joint is flexion and extension. This property enables different activities, from gait to ascending and descending stairs to squatting. Seen in Figure 2.3 are the different flexion-extension angles with activity examples. Besides these active activities, the knee joint also delivers stability.

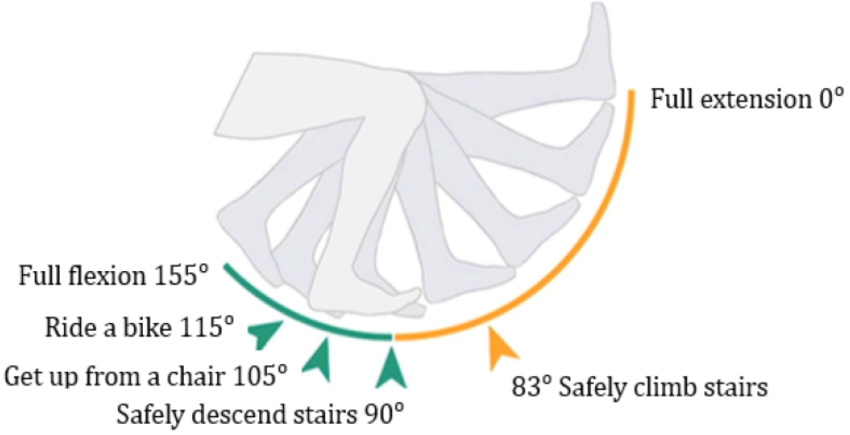

Figure 2.3: Flexion-extension of the knee joint paired with different activities (Ganesh, 2020a)

In addition to flexion and extension, the knee joint also controls other tibiofemoral motion. This motion is primarily restrained by the ligaments, as described above. The relevant tibiofemoral motions for this research are:

- AP translation: anterior-posterior translation of the tibia,
- IE rotation: internal-external rotation of the tibia,
- VV rotation: varus-valgus rotation of the tibia,
- SI translation: superior-inferior translation of the tibia.

In this report all motions are on the tibia, e.g. a external rotation is an external rotation of the tibia with respect to the femur. A visual representation of the above mentioned degrees of freedom can be found in Figure 2.4. A more in-depth look at what part of the knee joint has which restraining function can be found in table 1 of the work of Masouros et al. (2010).

2.2 ACL Lesion

This research focuses mainly on ACL lesions, as this is the most common knee injury (Majewski et al., 2006). An ACL lesion could be a stretched ligament, however also a partial or fully torn ligament. The ACL is responsible for 85% of the restraining force to limit the anterior translation of the tibia (Butler et al., 1980; Mader et al., 2007). Together with the PCL, the ACL also restricts internal and external rotation of the tibia (Ganesh, 2020a; Masouros et al., 2010).

Overcoming an ACL lesion can be done by ACL reconstruction (ACLR). This procedure requires surgery performed by an orthopaedic surgeon. As mentioned in the introduction of this report it involves removing the torn or partially torn ACL and replacing it with a donor ligament, known as an allograft, or with a tendon graft from another part of the knee, known as an

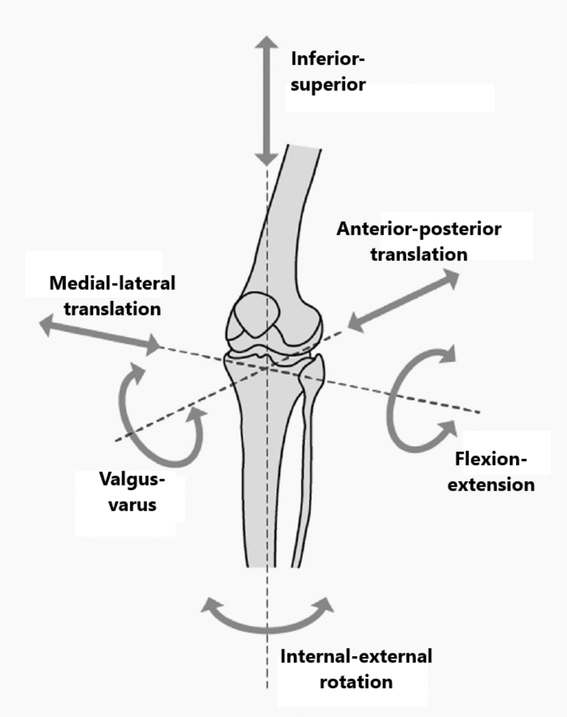

Figure 2.4: Biomechanical degrees of freedom knee joint. Altered by author, from Shenoy et al. (2013)

autograft (Mannel et al., 2004; Macaulay et al., 2011). An ACL reconstruction procedure comes with the risk of re-tearing the reconstructed ligament. The incident rate of a second ACL lesion within 24 months after ACL reconstruction with return to sport was nearly 6 times greater than that in healthy control participants (Paterno et al., 2014).

The use of a knee brace used post-operative has the benefit of significantly returning stability and symmetry to the knee but introduces limitations to the original motion of the knee joint (Lu et al., 2006). Mechanically the knee brace will also protect the graft from the forces it is subject to. The recommended use for a passive brace post-operative is 3 weeks, as longer use can have negative effects such as a contribution to atrophy (Chachula et al., 2012).

Another option of overcoming an ACL lesion is by means of physiotherapy. During this non-surgical approach, the patient is guided by an orthopedic specialist and a physiotherapist. The patient will follow a tailored rehabilitation program which mainly focuses on returning knee stability by strengthening the thigh muscles. During this rehabilitation, a (traditional) passive knee brace is worn by the patient to reduce the laxity caused by the ACL lesion. The brace facilitates reducing AP tibia translation and reducing the IE tibia rotation. Furthermore, the brace can improve proprioception, improving the sense of stability by the patient (Paterno, 2017).

2.3 ACL Lesion Biomechanics

An ACL lesion will alter the passive stability of the knee by reducing the restraining force caused by the ACL. The passive stability is affected in the anterior displacement, internal tibial rotation, and valgus rotation (Masouros et al., 2010). Furthermore, with an ACL lesion the active stability by the muscles is also diminished by the reduced proprioception (Dhillon et al., 2011).

As mentioned before the ACL plays a major role in the restraining force for anterior tibial translation. The total restraining force is produced for 85% by the ACL as shown by research by Butler et al. (1980) and Mader et al. (2007). The restraining force at 30° flexion was approximately 330N and at 90° flexion approximately 440N (Ganesh, 2020a).

The hamstring muscle plays a significant role in decreasing the increased anterior tibial translation. This is also the focus when opting for physiotherapy. Studies by Papadonikolakis et al. (2003) and Shelburne et al. (2005) showed by means of simulations that by increasing the isometric strength of the hamstring muscle the anterior tibial displacement could be reduced towards tibial displacement comparable to an intact ACL knee.

Shown by Williams et al. (2004) is that the quadriceps are not idle during flexion, when usually they are suppose to be. Such a involuntary movement is called dyskinesia. This introduces additional forces in the anterior direction of the tibia. This could be linked as a direct result of diminished proprioception. Another research by Williams et al. (2005) shows that in addition to the atypical muscle activity ACL deficiency causes quadriceps atrophy for most of the ACL deficient cases, weakening their quadriceps strength.

IE rotation is also affected by ACL deficiency. Research by Vassalou et al. (2016) shows that an increase in rotational laxity is an indication for ACL deficiency. Thus relating ACL deficiency to an increase in rotational laxity. This is further confirmed by the research performed by Naghibi Beidokhti et al. (2017b) in which it is shown there is an increase in rotational laxity when the ACL is dissected on three cadaveric knees.

2.4 Knee Brace Overview

Mentioned in Chapter 1.1 the use of a knee brace for ACLR and ACLD patients returns stability, symmetry and proprioception during rehabilitation. In the next two sections the (traditional) passive knee brace and the proposed robotic knee brace are elaborated.

2.4.1 Passive Knee Brace

A traditional knee brace is designed to stabilize the knee joint. These braces can be off-the-shelf or custom-fit for the patient and are available with different functions as goal. As described by Ganesh (2020a) there are four types of functions: prophylactic braces function as a protection of the knee during sports, functional braces support an already injured knee, rehabilitative braces function by limiting motion that would be harmful, and last the unloader/off-loader brace functions by relieving the load of the human body on the knee (in the superior-inferior (SI) direction). These braces are shown in Figure 2.5.

Beynnon et al. (1992) studied multiple functional braces representative both the typical custom-fit and off-the-shelf designs. In his work it is concluded that the braces tested (at 30° of knee flexion) were not able to reduce the strain on the ACL at AP loads over 100N. The traditional braces were able to reduce strain on the ACL at IE torsional loads up to 5Nm. A higher torsional load was not tested. In daily activities such as gait, the AP loads reached over 400N and IE loads over 13Nm according to analysis of OrthoLoad (2021)[3]. A similar AP loads for daily activities is mentioned by Ganesh (2020a).

[3] Specific analysis of gait at 4 km/h with a basic sport shoe of patient k1l. Specific file: `k1l_180908_1_15p`

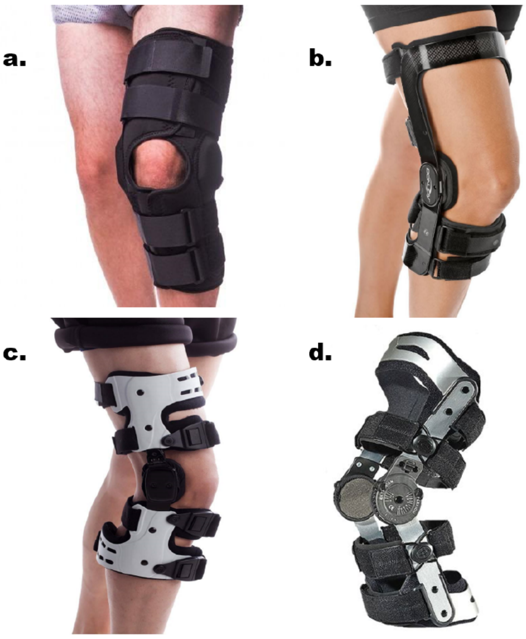

Figure 2.5: a. prophylactic brace (brand: BraceAbility), b. rehabilitative brace (brand: Donjoy Performance), c. unloader/off-loader brace (brand: Orthomen), and d. functional brace (brand: Carboflex). Image taken from Khambati (2019).

Another limitation of these passive braces is that they perform sub-optimal in non-weightbearing-to-weightbearing transition scenarios, mentioned by Beynnon et al. (2003) and Chachula et al. (2012). This transition happens during heel-strike and influences the human gait in daily life.

2.4.2 Robotic Knee Brace

This research report focuses on using the soft robotic knee brace depicted in Figure 2.7, actuated with PAMs, to counteract the laxity caused by ACL lesion in the AP and IE direction. These two directions are chosen as they are affected the most by an ACL lesion. The soft robotic knee brace can take of muscle activity, especially hamstring activity, to reduce the laxity caused by ACL deficiency.

The main improvement of the robotic brace over a traditional passive brace is that this research aims to lift certain restrictions and limitations from a traditional passive brace, such as hard limits in motion and increase support at larger loads. As mentioned in Section 2.4.1 a traditional passive brace is not able to reduce the strain on ACL in daily activities. It is thus also not capable of reducing the increased knee laxity caused by ACL deficiency.

Furthermore, the goal is to tailor the soft robotic brace more specifically to the patient. This can be done by setting custom restrains or state-specific forces that are tailored towards the rehabilitation program of the patient.

Example: The soft robotic brace can be set to allow a maximum AP displacement of 13mm at 90 degrees of knee flexion. Going over this threshold the brace could be able to, instead of a hard limit, increase the restraining force exponentially. This will gradually stop the motion and be more comfortable for the patient.

Another possibility is to upload healthy knee motion to the soft robotic knee brace. This way the controller of the soft robotic knee brace has a motion reference to track, and will keep the knee inside the brace within this healthy motion. This method is later described in Chapter 4. Because of this customization, the brace aims to result in a combination of a functional brace and rehabilitative brace and to be usable by ACLD and ACLR patients.

2.5 Pneumatic Artificial Muscles

Pneumatic artificial muscles (PAMs) are a type of soft actuators. These actuators work by contraction or extension as a result of changing the pressure inside the pneumatic bladder. PAMs are lightweight, are easily fabricated, not affected by electrical or magnetic interference, have similar load-length characteristics to biological human muscles, and are very compliant. The latter two make PAMs a suitable choice to use as an actuator that interacts with the human body.

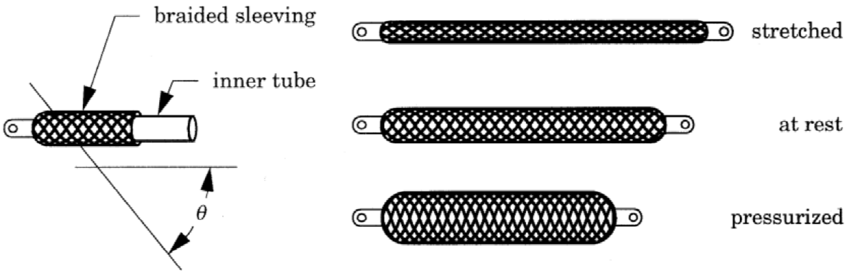

Figure 2.6: Pneumatic artificial muscles (Daerden and Lefeber, 2002)

The PAMs contract when positive pressure is applied, resulting in a force. PAMs can contract 25% to 40% of their non-pressurized length depending on their material and construction (Ganesh, 2020a). The contraction ratio ϵ can be derived using the length at rest ℓ_0 (non-pressurized) and the pressurized length ℓ, as shown in equation 2.1. Figure 2.6 shows a visual of both states.

$$\epsilon = \frac{\ell_0 - \ell}{\ell_0} \tag{2.1}$$

The force that a PAM is able to generate is based on multiple factors. The length ℓ, muscle diameter D, braid angle (also known as pitch angle or weave angle) θ, and the contraction ratio ϵ (Daerden and Lefeber, 2002). The equation for the force F is shown in 2.2.

$$F = \frac{\pi D_0^2 p}{4} \left(\frac{3}{\tan^2 \theta_0} (1-\epsilon)^2 - \frac{1}{\sin^2 \theta_0} \right) \tag{2.2}$$

With p the pressure supplied, D_0 the muscle diameter at rest, and θ_0 the braid angle at rest.

The limitations of using PAMs as actuators is that their actuation is slow and inaccurate compared to traditional rigid actuators (Alici, 2018). However, for this research the inaccuracy can be neglected as the actuator is used hybrid to a human joint and is insignificant with the inaccuracy induced by the human's motion.

2.6 Previous Work

2.6.1 Soft Robotic Knee Brace

This research focuses on implementing control on an already existing and researched robotic knee brace researched by Ganesh (2020a). Her research focuses on optimizing and improving the actuation system of the robotic knee brace developed by Khambati (2019), which is shown in Figure 2.7.

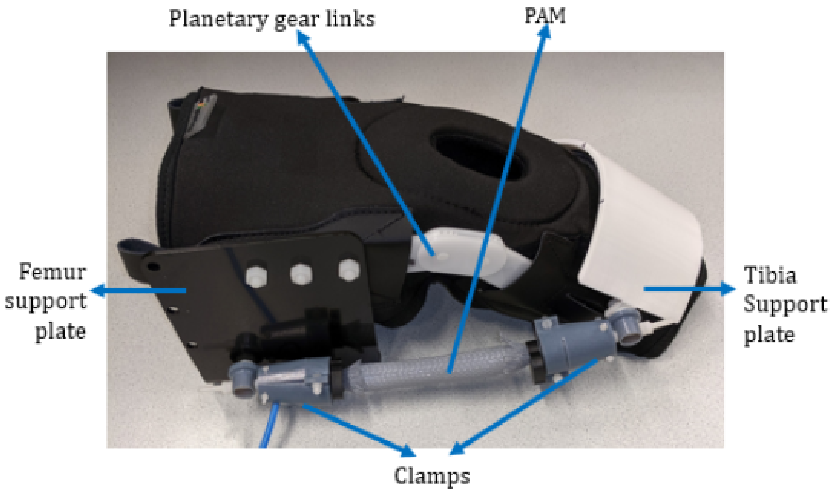

Figure 2.7: Soft robotic knee brace by Khambati (2019), taken from (Ganesh, 2020a)

The brace works by pressurizing the pneumatic artificial actuators attached to both sides (medial and lateral) of the brace. Pressurizing these applies a force posteriorly and partially relieves the hamstrings in a ACL deficient knee.

The knee brace by Khambati (2019) consist of a support plate for the tibia and femur, planetary gear links and the pneumatic actuators described above. These are integrated into a commercially available passive knee brace. The pneumatic actuators are used to connect the femur support plate and the tibia support plate. The main goal of Khambati (2019) was to reduce anterior translation of the tibia by assisting the hamstrings in a ACL deficient knee.

In the work of Ganesh (2020a) the limitations of the brace by Khambati (2019) were studied and improved upon. She mentions limitations such as the limited knee flexion angle, on which she improved the design from maximum flexion angle of 30° to 90°. In the work of Ganesh (2020b) the PAMs used in the robotic knee brace are further researched. The maximum actuator force of 80N has been taken from this research.

Another limitation is that the pneumatic actuators are only usable in a limited range. A single pneumatic actuator is not able to cover the whole 0° to 90° knee flexion angle range. A solution for this would be adding more pneumatic actuators, which leads to the issue of limited space on the knee brace. Currently, the attachment points of the pneumatic actuators are changed depending on the flexion angle of the knee.

Furthermore, the work of Ganesh (2020a) shows promising of recovering healthy internal and external rotation laxity of the tibia.

2.6.2 Knee Laxity for ACL Deficiency

Characterisation of ACL behaviour under different circumstances in this research is based on studies performed in (Naghibi Beidokhti et al., 2017a). This study is focused on different modelling strategies to improve finite element (FE) models of the human knee joint. During experiments laxity tests were performed on three human cadaver knees. The data obtained during these experiments will be used for characterization of the ACL intact knee behaviour of the ACL intact knee model in the research of this report.

The apparatus used to obtain the knee laxity is shown in Figure 2.8. The image is taken from another work of the same author. Here, flexion and extension of the knee is applied to the femur (marked with **f** in the figure). All other motion, such as internal-external (IE) rotation, anterior-posterior (AP) translation, valgus-varus (VV), and medial-lateral translation, are applied to the tibia (marked with **t** in the figure).

The data that was made available for this research was patient data C926B_R which is a right knee of a 63 year old male measuring 191cm tall and weighing 87.5kg.

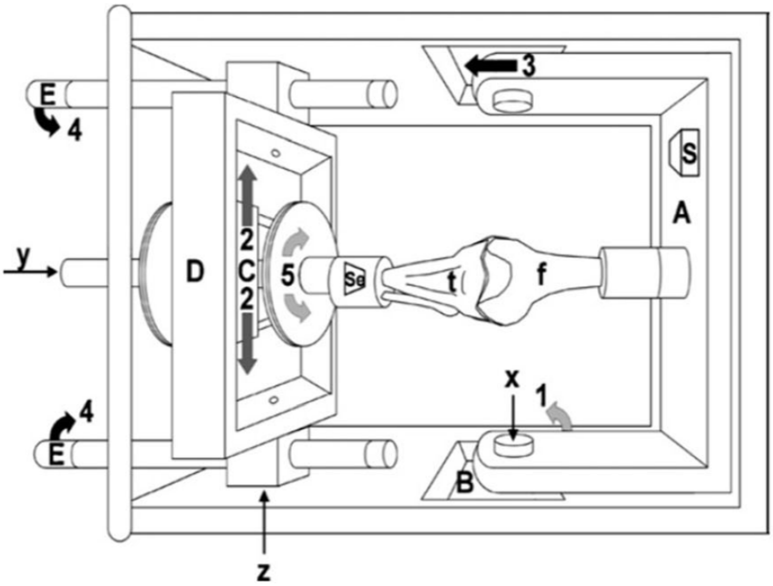

Figure 2.8: The six-DOF knee testing apparatus (Naghibi Beidokhti et al., 2017b)

The work of Naghibi Beidokhti et al. (2020) research focuses also on FE models of the human knee joint. Based on this research simulation results are obtained for ACL deficient laxity used to characterize the ACL deficient knee model. Due to the COVID-19 pandemic real world laxity tests were not possible, and the already existing data is used. How this data is exactly used is further explained in Chapter 3.

3 Simulation Model Design

3.1 Plant Introduction

For this research the original intention was to perform experiments in the physical lab and on cadaver knees. However, due to the COVID-19 pandemic and the closure of the most of the physical labs in the Netherlands the choice has been made to focus on creating a model of the knee joint on the data available from previous studies and experimenting with the said knee model. The knee model is specifically made for this research.

A multi-body physical system model of the left knee has been created in Simulink using Simscape, a multidomain physical systems modelling and simulation extension. The model allows the necessary motions needed for this research: flexion and extension of the knee, AP translation of the tibia and IE rotation of the tibia. A extension has been made for VV motion, but not used to reduce complexity.

The requirements for this research is that the model is able to accurately represent AP and IE laxity at different knee flexion angles corresponding to the experiments in Section 2.6.2. It must then be possible to exert forces, by the soft robotic brace actuators or by external disturbances, on the model to which the model must result in motion also according to the experiments in Section 2.6.2. For this reason the way the model behaves represents the laxity testing apparatus.

The model created in Simulink is dynamic, it includes masses, damping, etc. However because the laxity data is of static nature the model is considered to be static. The dynamical effects of the mass of the solids and small damping of the joints is neglected. This is possible as the motion of considered to be slow. For this reason a static model is sufficient for this research.

3.2 Plant Overview

The model is based on the kinematic sketch in Figure 3.1, which in turn represents the laxity testing apparatus. This way the data obtained from the real world laxity experiments could be incorporated into the model in a representative way.

The kinematic sketch can be interpreted as in which the femur is constrained and the tibia is set in different knee flexion angles. When loads are applied to the tibia will behave as a rigid beam which rotates around the ankle joint, motion comparable to the laxity testing apparatus. The tibia is depicted in two configurations: posterior translated (pink) and anterior translated (orange). The dotted line indicates a zero translation in the AP direction.

The tibia is segmented into two rigidly connected beams with a 1-DoF rotation about the axis running in the length of both segments. This is done for a more practical simulation of IE rotation. Further explained in Section 3.3.3. The decrease in rotational inertia as a result of this segmentation can be neglected for the application of this research due to the slow motion.

In the rigid body model the femur and tibia are modeled as beam elements, solids, with inertial properties based on its geometry and mass. The density is assumed to be uniform for simplicity. Table 3.1 show the dimensions and masses of the elements. These properties are added as an indication and will not affect simulation results significantly.

Solid	%BW	Mass (kg)	Geometry ($l \times w \times h$) (cm)
Femur	14.16	12.39	45 × 3 × 3
Tibia Proximal	3.09	2.57	25 × 3 × 3
Tibia Distal	1.24	1.03	10 × 3 × 3

Table 3.1: Femur and tibia properties. Percentages from (ExRx.net, 2021). Masses for the segmented tibia is according to geometry ratio. Bodyweight (BW) from Cardavic knee identifier: C926B_R 87.5 kg

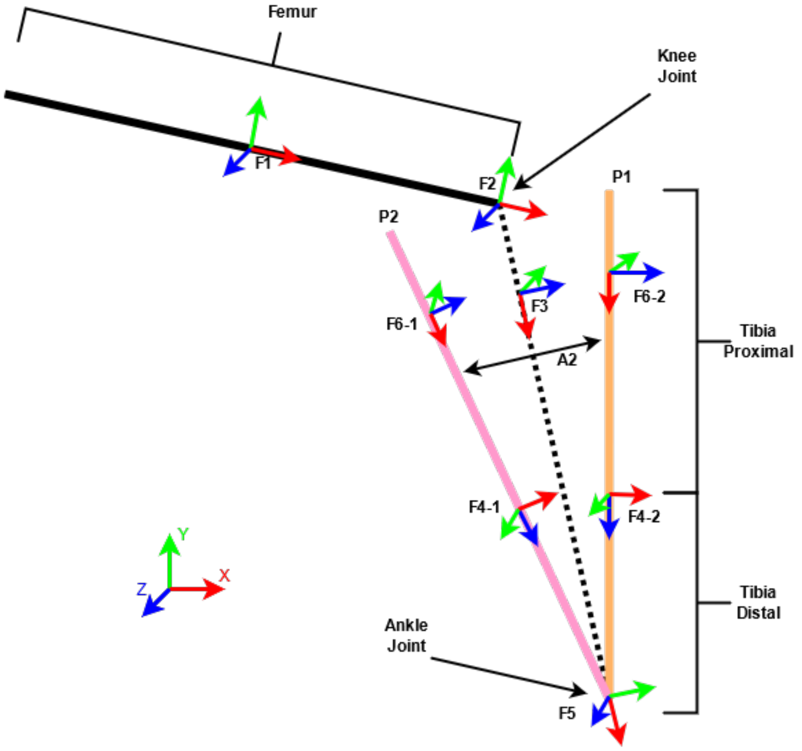

Figure 3.1: Kinematic sketch of the knee multi-body system with the most important frames attached

Figure 3.2 shows how the kinematic sketch in Figure 3.1 is converted into Simulink multi-body physical system. The system is divided in three areas: The knee model (blue area) contains the femur and tibia as three beam element solids and four subsystems that allow motion and add characteristics. The pneumatic artificial actuators (green area) contain the actuators that represent the brace. This is explained in Section 3.4. Lastly, the external forces and disturbances (violet area) contains the system responsible for applying external forces and disturbances explained in more detail in Section 3.5.

Figure 3.3 visualizes the solids and subsystems of the knee model in the mechanical representation of the model. It also includes both medial and lateral PAMs.

3.3 Plant Subsystems

The plant contains four subsystems. The subsystem Knee Joint (Section 3.3.1) sets the flexion and extension of the model, this can be a discrete flexion angle or a continuous flexion angle profile. The subsystems Ankle AP and Ankle IE (Section 3.3.3) constrain both parts of the tibia to rotate around the ankle joint in the AP and IE direction only. Last, the subsystem AP & IE

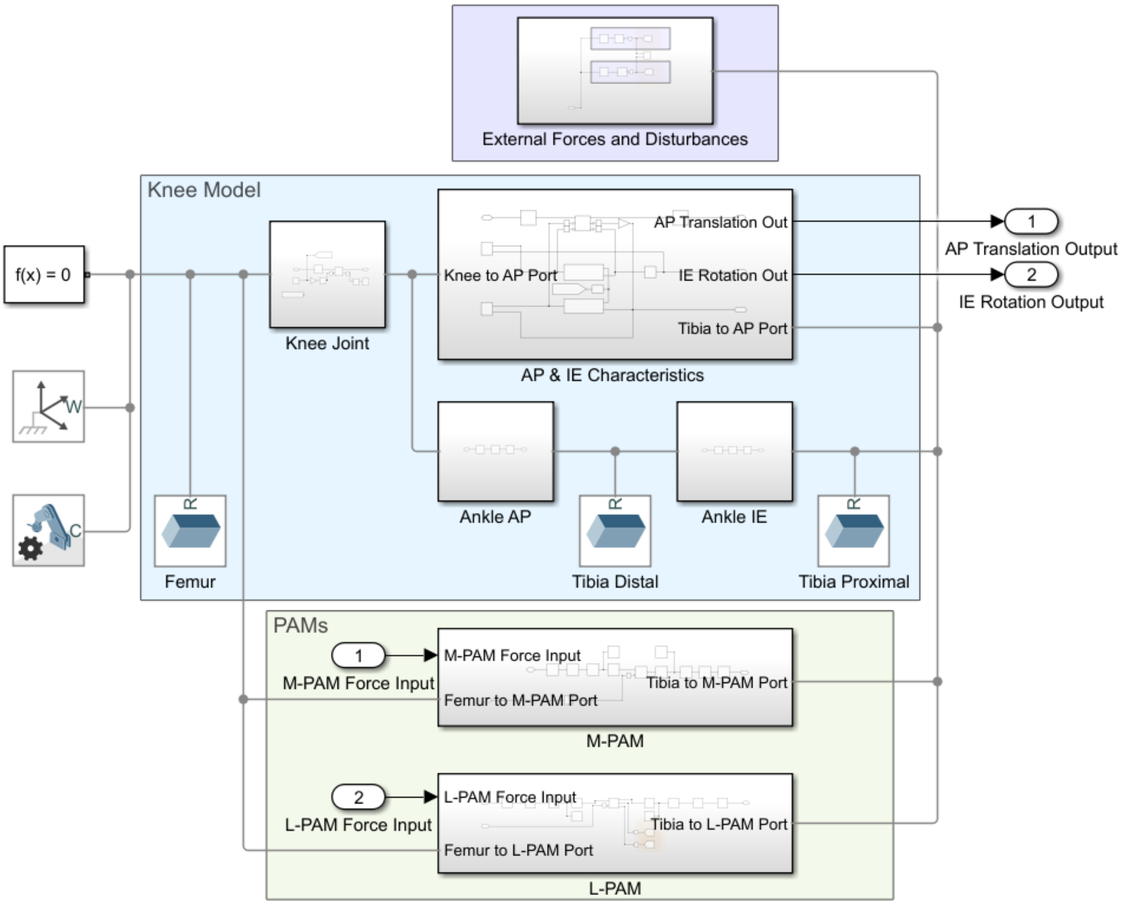

Figure 3.2: Overview of the components of the plant in Simulink

Characteristics (Section 3.3.4) add laxity characteristics to the model. All subsystem can be found in the Simulink overview in Figure 3.2.

3.3.1 Knee Joint

The Knee Joint subsystem represents the knee of the human body. It is modelled to allow flexion and extension in the simulation. The subsystem connects the femur solid with the tibia solids and allows a 1-DoF rotation between them that represents the flexion angle.

In Figure 3.4 the expanded view of the subsystem can be seen. From left to right, the *Femur to Knee* block is a rigid transform block. Here the femur solid centre frame (F1 in Figure 3.1) is rigidly translated to the position of the knee joint frame (F2), so rotation takes place where femur and tibia meet.

The *Knee Joint* block is a revolute joint block and allows a 1-DoF rotation around the z-axis. This rotation represents the knee flexion and its value is provided by the *Knee Angle Setpoint* block in degrees, this can either be a discrete knee flexion angle or a continuous knee flexion angle profile. The knee flexion is made globally available throughout the system by the *kneeAngleTag*. The location of the Knee Joint is approximately indicated in Figure 3.3.

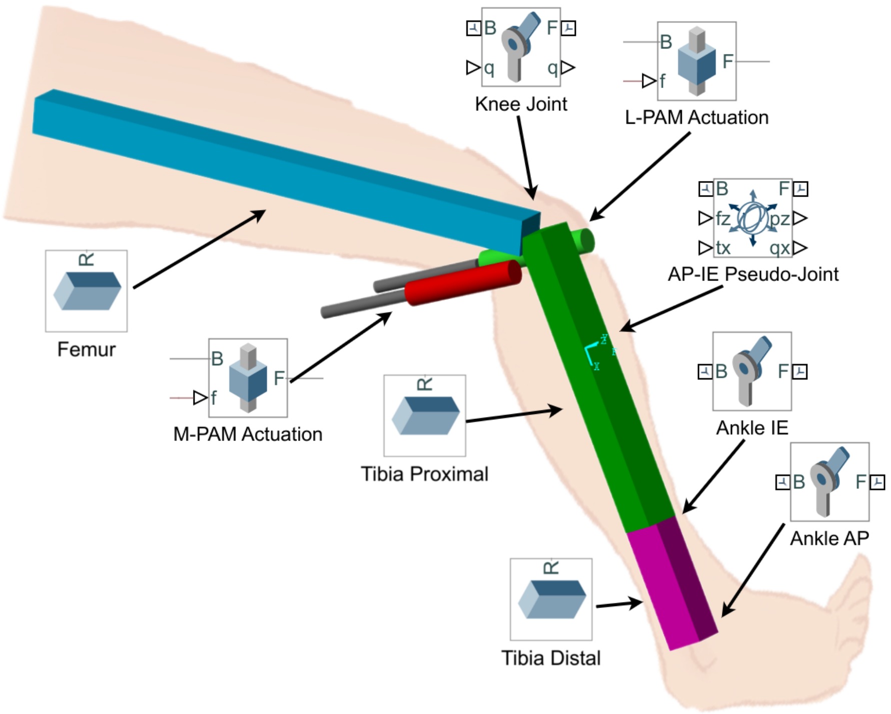

Figure 3.3: Visual mechanical representation of the Simulink model in a flexion angle of 30 degrees. The solids and subsystems are indicated.

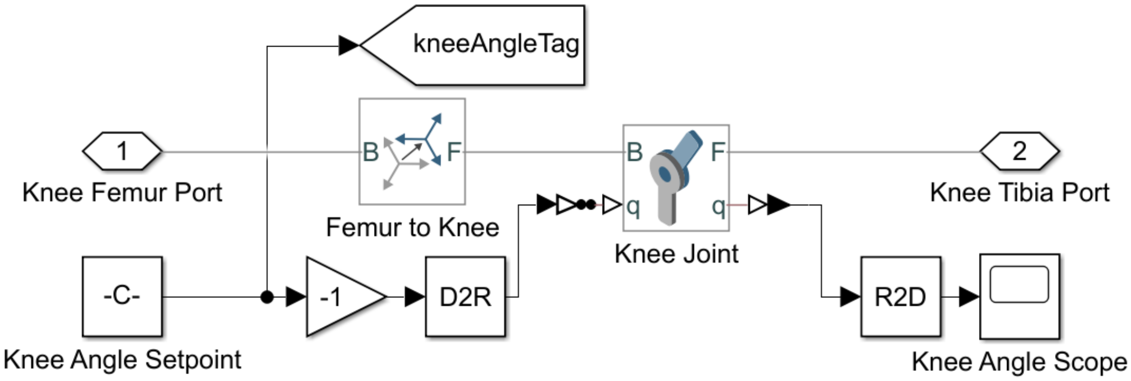

Figure 3.4: Knee Joint subsystem

3.3.2 Knee Flexion Angle Profile

As mentioned in the previous section, the setpoint of the knee flexion angle can either be a discrete value of 0°, 30°, 60°, or 90°, or can be a time dependent continuous signal. For simulating gait a time dependent continuous, changing, knee flexion angle is required, also referred to as continuous knee flexion angle in this report. The knee flexion angle during gait has been obtained from Bovi et al. (2011). The data is labeled as gait at natural speed of an adult. For the purpose of this research this is sufficient. Figure 3.5 shows the knee flexion angle (bottom) of a single cycle of gait with their corresponding gait phases (top).

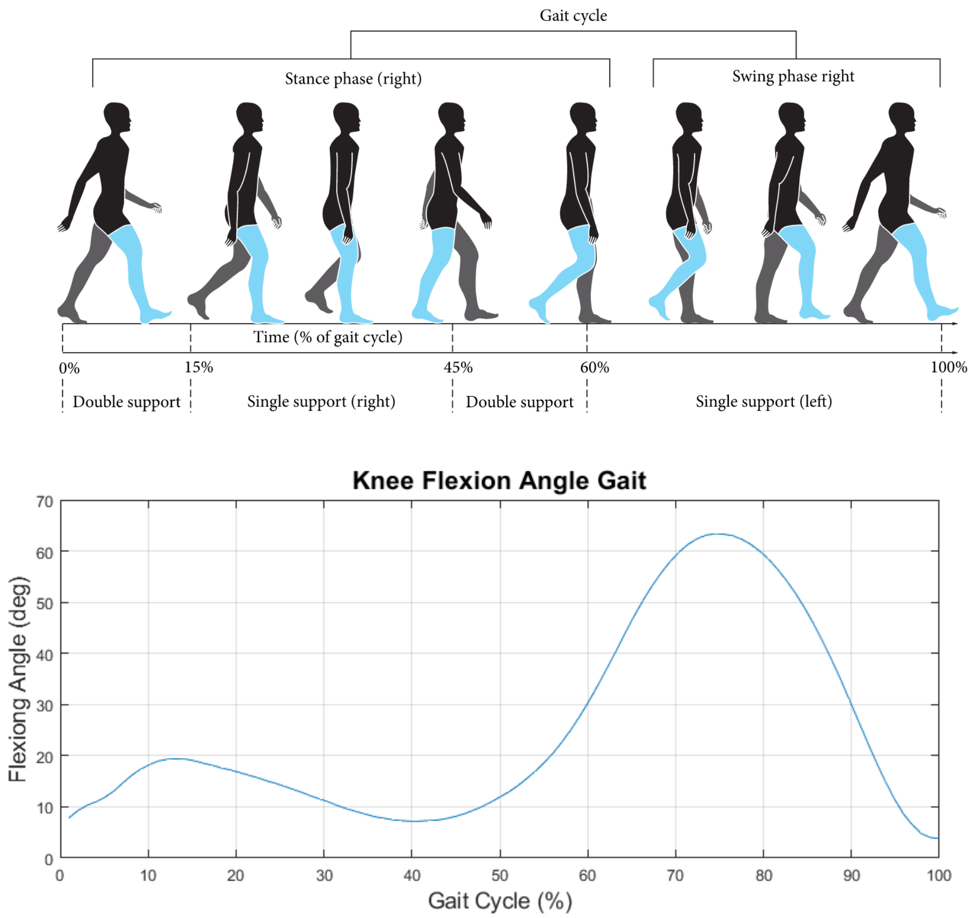

Figure 3.5: Knee flexion gait cycle. Top image taken from Dhillon et al. (2011)

3.3.3 Ankle AP & Ankle IE

In the subsystems Ankle AP and Ankle IE some modelling techniques are used to make modeling easier and allow for a more constrained motion closer to the laxity experiments in Section 2.6.2. The knee joint frame (F2) is directly connected to the ankle frame (F5), visualized by the dotted line in Figure 3.1. To this rigid connection no mass (solid) is connected. From the ankle frame the tibia (distal and proximal) frame(s) are connected, and to those frames their solids.

Figure 3.6 shows expanded view of the Ankle AP subsystem. *Knee to Ankle AP* is the rigid translation from the knee frame (F2) to the ankle frame (F5). To this frame a 1-DoF revolute joint is connected which allows the (pivot) motion of the tibia in the AP direction by rotating around the ankle joint. This motion represents the dorsiflexion and plantarflexion of the ankle but the ankle is constraint and the tibia moves. The rotated frame is then transformed, with the *Ankle AP to Tibia Distal* block, to the centre frame of the tibia distal which connects to its corresponding solid.

Moving more proximal the Tibia Distal and Tibia Proximal solid meet at frame F4. At this point the two solids can rotation relative to each other around the axis in the length of both tibial solids by means of the Ankle IE subsystem.

Figure 3.7 shows the expanded view of the Ankle IE subsystem. In a similar manner as the Ankle AP subsystem, first the tibia distal centre frame is transformed to frame F4 by the *Tibia Distal*

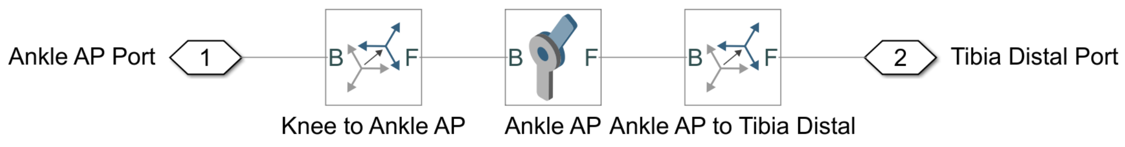

Figure 3.6: Ankle AP subsystem

to Ankle IE block. Here a 1-DoF revolute joint allows for the IE rotational motion. This joint is perpendicular to the joint used in the Ankle AP subsystem. Combined they allow for AP and IE motion. The resulting rotated frame is rigidly transformed to the centre frame of the tibia proximal solid.

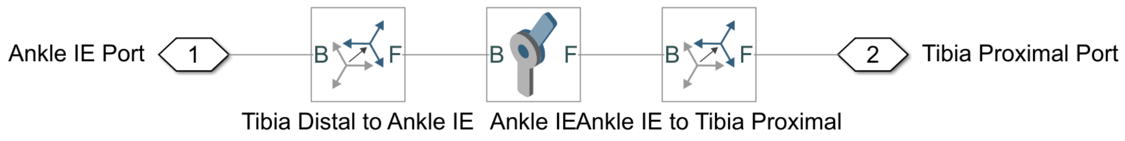

Figure 3.7: Ankle IE subsystem

3.3.4 AP & IE Characteristics

The AP and IE laxity characteristics are introduced in the AP & IE Characteristics subsystem. This subsystem can be considered a pseudo-joint. The sole purpose of the joint is to add the laxity characteristics from the laxity data from the studies performed in Section 2.6.2. One could see this pseudo-joint as a translational and rotational spring between the actual tibia and the zero-position (zero translation and zero rotation, visualized by the dotted line in Figure 3.1) of the tibia.

Figure 3.8 shows to expanded view of the subsystem. From top-left to top-right: the *Knee to AP* rigid transform block transforms frame F2, the tibia side of the knee joint, along the dotted line in Figure 3.1 to frame F3. Here one side of the 6-DoF Bushing Joint block, called *AP-IE Pseudo-Joint*, is attached. The other side this bushing joint, frame F6, is connected to the centre of the Tibia Proximal solid frame with rigid transform block *Tibia to AP*. The location of frame F3 corresponds to approximately where the forces are applied in the real world laxity tests.

The AP and IE laxity characteristics is added by supplying the joint block with a displacement and rotation-depended external force and torque. Both force and torque are spring-like forces and depend on the state of the joint block, namely the translation along the z-axis for the force and the rotation about the x-axis for the torque. These external force and torque are explained in the following sections.

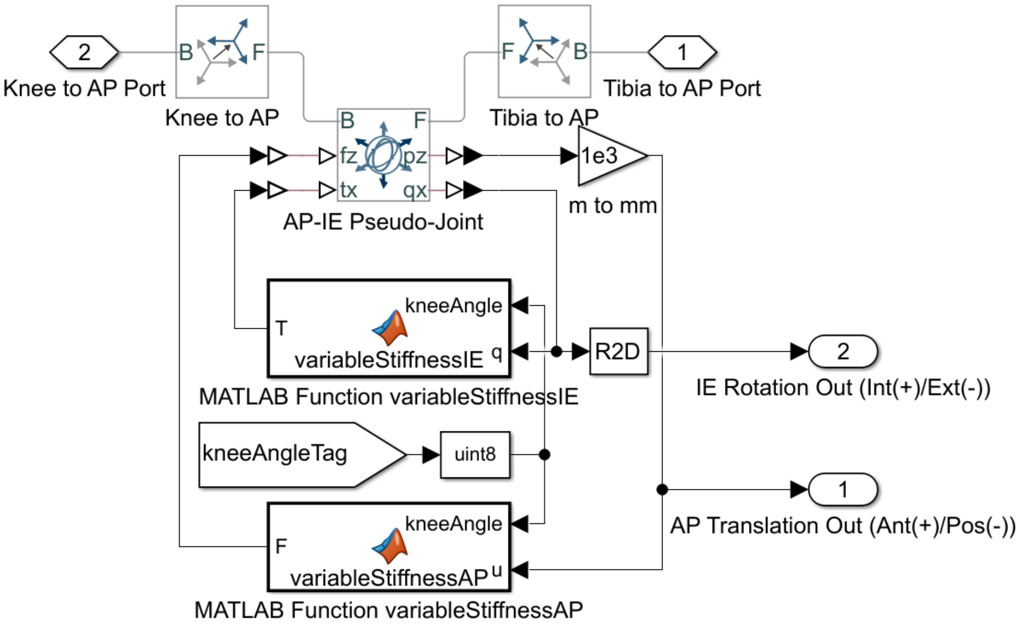

Figure 3.8: AP & IE Characteristics subsystem

3.3.5 AP Stiffness

The AP stiffness supplied as external force to the input (fz) of the AP-IE Pseudo Joint is computed using a MATLAB Function block, named *MATLAB Function variableStiffnessAP* in Figure 3.8. This MATLAB function block has two inputs and one output. The block receives the AP displacement, in mm, from the AP-IE Pseudo Joint block and receives the current flexion angle of the knee from the globally available *kneeAngleTag*. The block output of the block is a force which is applied along the negative z-axis of the base frame of the AP-IE Pseudo-Joint to act as a spring-like force.

A note here is that because of how the tibia rotates around the ankle joint in this model the centre of the follower frame of the AP-IE Pseudo-Joint moves with an arc-motion when the angle of the ankle is changed. The z-axis of the base does not follow this arc-motion, but the displacement is measured along the z-direction of the base frame. For small arcs this difference is insignificant and is neglected in this research.

The variableStiffnessAP MATLAB function can be found in appendix H. The function receives the current knee flexion angle and current displacement. It will then call the function `APForceFunc` which returns the correct spring force at those two variables. This function is based on a curve or surface fit that will be discussed in the next sections. The code for this function can also be found in appendix H.

3.3.5.1 AP Laxity Data

The characteristics of the AP motion is based on fitted data, which is a 2D curve for discrete (non-changing) knee flexion angles or a 3D surface fit for continuous (changing) knee flexion angles. Besides these two types of fits, the data that is used comes from two sources: data from real world laxity experiments on a cadaver knee and data from finite element simulations, as mentioned in Section 2.6.2.

The correlation between AP force and AP motion is estimated to be a monotonic cubic polynomial. This cubic shape is confirmed the works of Beynnon et al. (2003). The ACL deficient data set only contains the data for the two extreme forces. Due to the limited data, a cubic shape cannot be observed nor can be fitted. However, a monotonic cubic shape is expected. An increase in laxity caused by ACL deficiency decreases the restraining forces but other restraining forces from e.g. the tissue surrounding the knee stay unchanged. This way the general shape of the laxity should remain the same, but the laxity should increase.

A solution to this is based on how an ACL lesion changes the laxity of the knee. According to Masouros et al. (2010) the ACL is primarily restraining the anterior motion of the tibia. The posterior motion is not restrained by the ACL. This means that an ACL lesion does not, or insignificantly, affect posterior laxity. This is further confirmed by comparing the ACL intact knee laxity data to the ACL deficient knee laxity data. It can be seen that the difference between ACL intact and ACL deficient posterior extreme is significantly smaller than the difference at the anterior extreme. This can be seen in appendix B.

Because of this property and the assumption that the displacement crosses the same zero-force point, the data of the laxity test of a ACL intact knee and the laxity results from finite simulation of an ACL deficient knee can be augmented. The posterior laxity has been taken from the ACL intact knee laxity data set, and the anterior laxity has been taken from the ACL deficient knee simulation data set. This method increases the number of data points and allows a cubic degree behavior be seen. To this a monotonic cubic function can then be fitted. This result is further used for fitting ACL deficient knee characteristic curves and surfaces. The ACL intact, ACL deficient, and augmented laxity data sets are visually depicted in appendix B.

3.3.5.2 AP Discrete Flexion Angle Curve Fit

For discrete knee flexion angles 2D curve fits are used in the `APForceFunc` described in Section 3.3.5. This 2D curve fit is based on the augmented ACL deficient knee data described in Section 3.3.5.1. This is done for the four individual discrete knee flexion angles, resulting in four AP 2D curve fits. These four fits are later used to create a surface fit.

The augmented data is imported into MATLAB and using the Shape Language Modeling (SLM) by D'Errico (2017) toolbox a 2D piecewise polynomial fit is obtained. The reason for using the SLM toolbox as it allows to set additional parameters, in this case it is necessary to create a monotonic function as output as it reflects real life more accurately.

The resulting 2D curve fits can be seen in Figure 3.9 with their fitting specifications in Table 3.2. Notice that the fits are extrapolated, this is necessary for the purpose of this research as forces applied to the knee joint are generally larger than 99.8N.

The 2D curve fits can be used in simulations involving discrete knee flexion angles of 0°, 30°, 60°, and 90°. To simulate gait with a changing continuous knee flexion angle a 3D surface fit, this is explained in the next Section 3.3.5.3. The MATLAB code for these discrete fits can be found in appendix G.

Flexion Angle	RMSE AP (N)	RMSE IE (Nmm)
0°	0.86	1583
30°	2.31	461
60°	1.88	410
90°	1.31	28

Table 3.2: Root Mean Square Error of all discrete 2D curve fits for the AP and IE augmented data

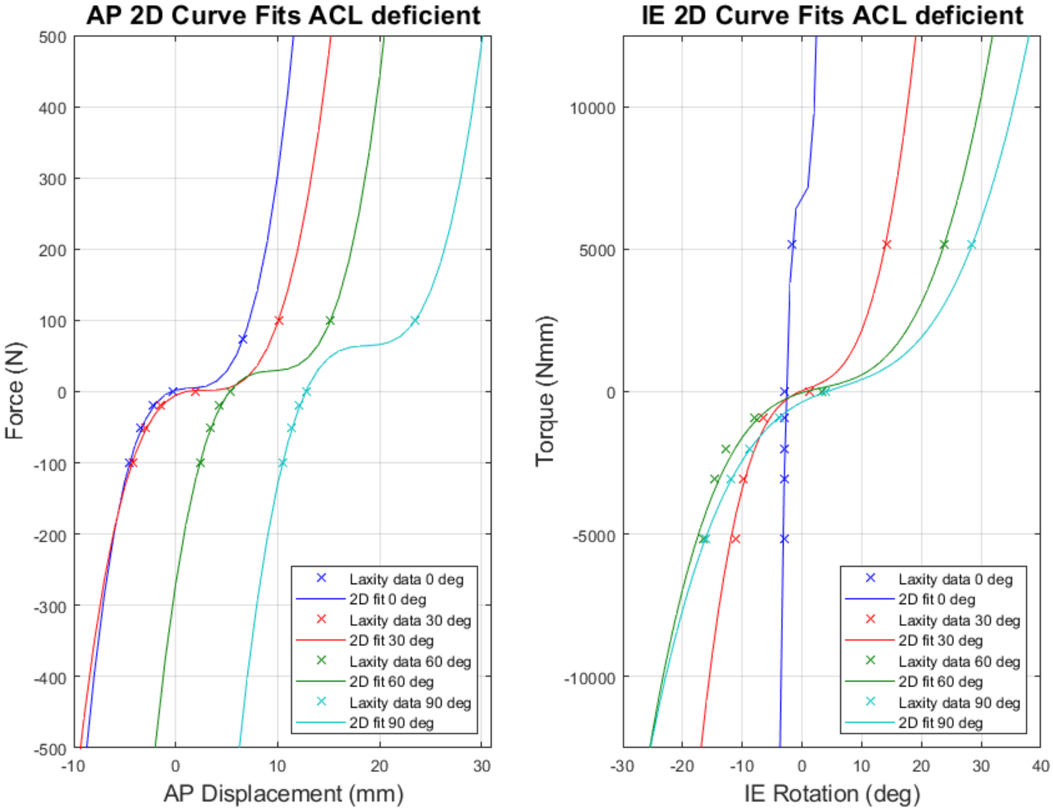

Figure 3.9: 2D curve fits for augmented ACL deficient data to be used in discrete knee flexion angle simulations. AP 2D curve fit and augmented laxity data left, IE 2D curve fit and augmented laxity data right.

3.3.5.3 AP Continuous Flexion Angle Surface Fit

As mentioned in the previous section, to simulate gait and thus continuous changing knee flexion angles it is necessary to obtain a smooth surface based on the augmented ACL deficient laxity data. Unfortunately the SLM toolbox does not allow surface fitting, setting custom parameters such as forcing monotonicity is not possible. A workaround is to use the discrete curve fits, obtained in Section 3.3.5.2, which are monotonic. From these fits a high number of data points is extracted (approximately 100). This is done to force monotonicity when fitting with MATLAB's own `cftool`.

Importing those extracted data points of all four discrete knee flexion angles into `cftool` allows us to fit a surface. To prevent unwanted behaviour it has been decided to use a linear interpolant between the knee flexion angles. This might not reflect real life to the best extend, but is a safer fitting option to prevent over- and under fitting. This also forces the monotonicity from the discrete curve fits. The resulting fit can be seen in Figure 3.10. This resulting surface can be used to determine the stiffness of the AP direction at any flexion angle in the range of 0 to 90 degrees. The MATLAB code for the surface plot can be found in appendix G.

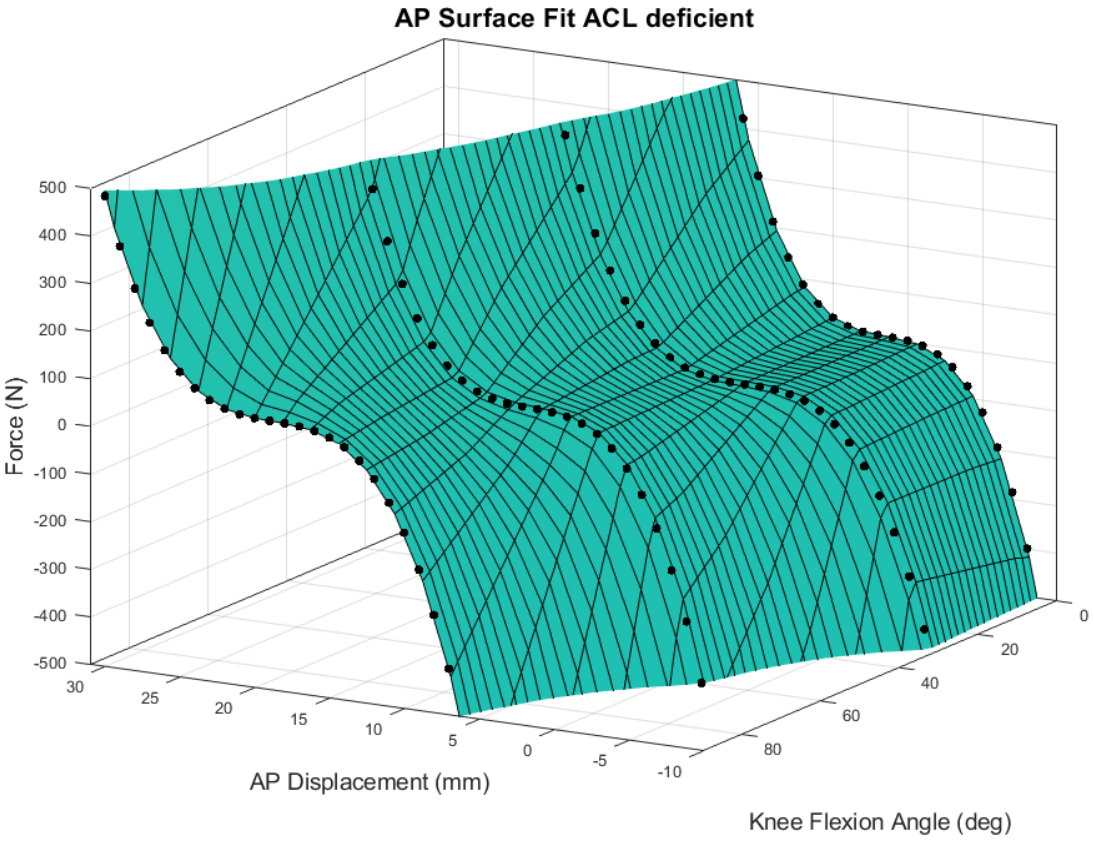

Figure 3.10: Resulting AP displacement surface plot

3.3.6 IE Stiffness

The IE stiffness is implemented in a similar manner as the AP stiffness. Based on the current IE rotation and knee flexion angle a torsion spring force is computed based on fitted data by the MATLAB Function block *MATLAB Function variableStiffnessIE*. The output is applied as external torque about the negative x-axis of the 6-DoF bushing joint.The variableStiffnessIE MATLAB function can be found in appendix I.

3.3.6.1 IE Laxity Data

The IE motion characteristics have been implemented in a similar manner as the AP motion described in Section 3.3.5.1. The IE motion is based on fitted data in the form of a 2D curve for discrete knee flexion angles or a 3D surface for continuous knee flexion angles. Similar to the AP motion, the data used comes from the same two sources: data from real world laxity experiments on a ACL intact cadaver knee and data from ACL deficient finite element simulations.

The correlation between IE torque and IE motion is estimated to be a monotonic cubic polynomial. The graphs can be seen in appendix B. With the IE motion from the ACL deficient finite element simulations the same issue is present as with the AP motion, only two data points are available at both extremes. Here, a monotonic cubic shape is expected for the IE motion like the ACL intact laxity data.

Using the same methodology as described for the AP displacement ACL deficient curve, it is possible to combine ACL intact and ACL deficient data to increase the number of data points on which a curve can be fitted. The research of Masouros et al. (2010) shows that the ACL has a restraining function in the internal tibial rotation, but not in the external tibial rotation. Using this information, the external rotational laxity data can be used from the ACL intact data set. The data point for internal rotation is taken from the ACL deficient data set. This allows for an augmented data set to which a monotonic cubic function can be fitted, also shown in appendix B.

3.3.6.2 IE Discrete Flexion Angle Curve Fit

For the discrete, non-changing, knee flexion angle case four 2D curves are fitted to the laxity data described in Section 3.3.6.1. To each individual knee flexion angle data set a monotonic (piecewise) polynomial is fitted. The resulting polynomial should be representing the real world behaviour of the ACL deficient knee, meaning it should be shaped as a cubic monotonic polynomial.

Similar workflow has been applied as in Section 3.3.5.2. The augmented data is imported into MATLAB and using the SLM toolbox a monotonic cubic shaped 2D piecewise polynomial fit is obtained. These fits can be seen in Figure 3.9 and its corresponding goodness of fit can be seen in Table 3.2.

These 2D curve fits are then used standalone in discrete knee flexion angle simulations, or for creating the surface plot as explained in the next section. The MATLAB code for this can be found in appendix G.

3.3.6.3 IE Continuous Flexion Angle Surface Fit

For continuous knee flexion angles the surface fit for IE rotation has used the same methodology as its AP variant in Section 3.3.5.3. The discrete curves from Section 3.3.6.2 are evaluated at a high number of points to force the surface fit to follow these curve fits.

After, the evaluated data points are imported in MATLAB's `cftool` to which a surface is fitted using linear interpolation. The resulting fit can be seen in Figure 3.11. This surface can be used to determine the IE stiffness at any flexion angle in the range of 0 to 90 degrees of knee flexion. The MATLAB code can be found in appendix G.

3.4 Modelling the Robotic Brace

The soft robotic brace is added onto the knee model. The soft robotic brace is modelled as two actuators on the medial and lateral side of the knee. Figure 3.2 showed the implementation of the PAMs in the Simulink model in the green area.

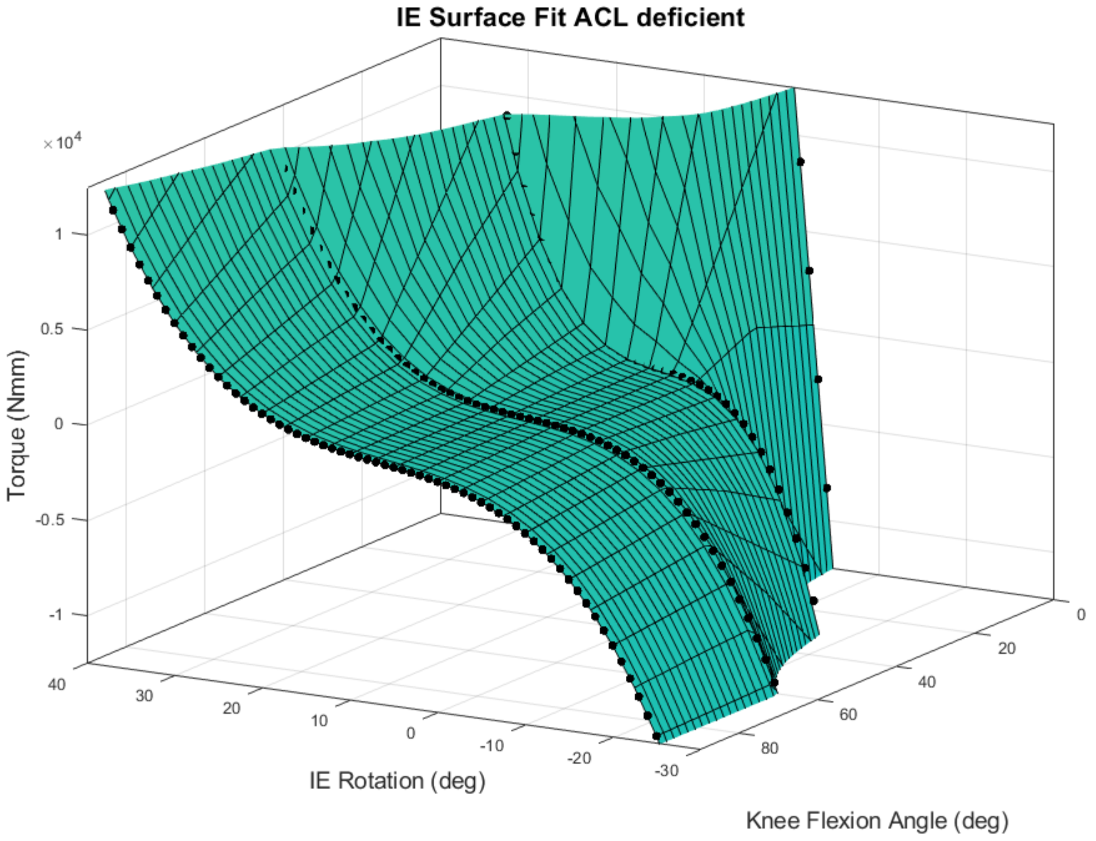

Figure 3.11: Resulting IE rotation surface plot

3.4.1 Pneumatic Artificial Muscles

The PAMs are modelled as ideal force controlled prismatic joint. The force is provided by the control signal of the controller in Chapter 4. The prismatic joint will exert this force between the femur attachment point and tibial attachment point. These points are further described in the next section.

As the known characteristics of the PAMs used are limited to the maximum actuation force of 80N the PAMs are modelled as a ideal prismatic joint actuator with force input. To add the slowness of the PAMs a slew limiter is added which allows setting the maximum rising and falling skew rate. By setting this rate a PAM can be modelled in a very simplistic manner. The Simulink implementation of a PAM is shown in Figure 3.12.

Figure 3.3 shows the knee model with both the medial and lateral PAMs in mechanical view.

3.4.2 Connection to the Knee Plant

The PAMs are rigidly connected to the femur and to the tibia. In the real world this would be the femur support plate and the tibial support place, however in this model it is to be assumed to be rigidly connected to the bones and not take into account the nonlinear skin interaction. The modelling decision is made to make the model less complex.

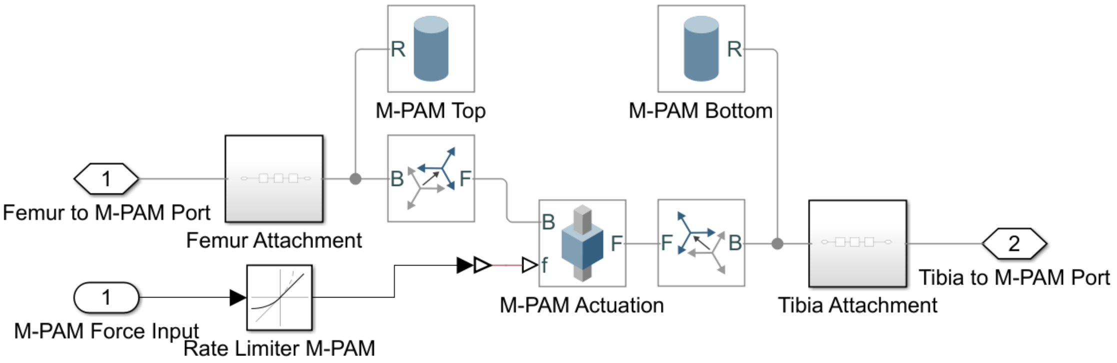

Figure 3.12: Simulink implementation of a pneumatic artificial muscle. Shown is the medial actuator. The lateral actuator is identical.

The research by Ganesh (2020a) shows a crossed configuration and a parallel configuration of the PAMs, shown in Figure 3.13. The research also shows optimal configuration at each flexion angle by means of a weighting factor. Based on this information it has been decided to use the following configuration.

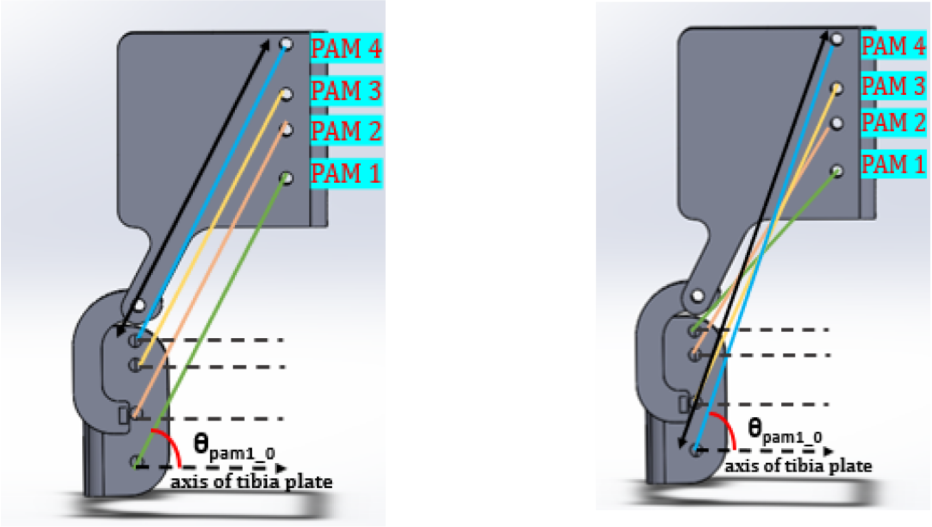

Figure 3.13: Pneumatic artificial muscles in parallel (left) and crossed (right) configuration (Ganesh, 2020a)

For discrete knee flexion angles it has been chosen to use a mix of individual configurations, as only one PAM (per side) is present on the brace at each time this is allowed. From the parallel and crossed configuration the optimal PAM position has been chosen from both configurations resulting in the following:

- 0 degrees of knee flexion: PAM 4 from the crossed configuration,
- 30 degrees of knee flexion: PAM 1 from the crossed configuration,
- 60 degrees of knee flexion: PAM 4 from the parallel configuration,
- 90 degrees of knee flexion: PAM 4 from the parallel configuration.

For continuous knee flexion angles a single PAM is chosen, as during gait it is impossible to switch positions of the PAMs. For this PAM 4 in a parallel configuration is chosen as it was the most optimal in the range of 0 to 90 degrees of knee flexion.

For the implementation in the Simulink model it is assumed that the femur and tibia are parallel to the attachment points in the respective support plate and that they are positioned in line with the middle of the rotational joint of the brace that allows the flexion and extension.

3.5 External Forces (and Disturbances)

External forces (and torques) can be applied to the model as shown in Figure 3.2 in the violet area. Here the AP force is applied at frame F3, shown in Figure 3.1.This is used to check the validity of the plant compared to the laxity data. During laxity test the AP forces are applied approximately at this spot, 10cm below the joint. The same goes for the externally applied torques.

Using this method the external forces from a gait cycle can be applied to the model as well. For this the location is modified to represent the location of measurement from (OrthoLoad, 2021). This location is located at the height of the lowest part of the polyethylene insert of the knee implant, the exact location is not know so in the model is it assumed to be at the most proximal part of the tibia. For applying IE torques during gait the IE axis is rotated backwards by about 7 degree relative to the length axis of the tibia bone. This corresponds to the tibial implant component used in (OrthoLoad, 2021).

The forces from OrthoLoad (2021) are shown in Figure 4.3 and further described in Section 4.3 where they are also used to generate a motion reference signal.

3.6 Model Validation

The Simulink knee model is validated for two cases. The first being the use while in discrete knee flexion angle. This is done at the knee flexion angles of the laxity experiments in Section 2.6.2 (0°, 30°, 60°, and 90°). Here laxity data from experiments is compared to the output of the model.

In the second case, the output of the model is also validated at knee flexion angles and forces and torques not performed during the laxity experiments. In addition to the four knee flexion angles, the model has also been validated at 15°, 45°, and 75° of knee flexion. These angles are tested to verify the workings of the surface fit as these three additional knee flexion angles cannot be compared to real world date. Of which is no laxity data available. The extrapolated forces and torques are also validated to check on realistic behaviour, however not validated against real world data as no laxity data available at these loads.

The validation is performed on the model with real world ACL intact laxity fitted data or ACL deficient laxity fitted data applied to the model characteristics, see Section 3.3.4. The ACL intact validation is performed to compare the model more closely to the real world, as for ACL deficient little data is available. The ACL deficient validation is then performed to show the difference in characteristics and if the laxity is increased as expected.

To obtain the output AP displacement and IE rotation the same forces and torques are applied to the model which were applied to the real world laxity test. These are applied as external forces with the subsystem described in Section 3.5.

3.6.1 Validation with Real World Data

Figure 3.14 shows the AP displacement validation for an ACL intact knee model (left) and ACL deficient knee model (right). It can be observed that the model follows the real world laxity data closely. Especially the shape is as expected. When comparing the ACL intact to the ACL deficient AP displacement it can be seen that in the posterior (negative displacement) not a lot is changed, as expected as the ACL intact data is used for that part. Looking at the anterior displacement a significant increase in displacement is observed, which represents an increase in laxity.

Figure 3.15 shows the IE rotation validation for the same two knee models, ACL intact (left) and ACL deficient (right). The same can be seen for the IE rotation as for the AP displacement. The model follows the real world laxity data closely, and again the shape of the model is representative of the real world. For the comparison between ACL intact and ACL deficient an increase in laxity can be seen in the internal direction (positive rotation), as expected as the external direction is based on ACL intact data (see Section 3.3.6.1).

The MATLAB code and data for the AP and IE ACL intact and deficient data fit can be found in appendix F and G. Table 3.3 shows the error metrics of between the real world laxity and the model validation output for the ACL intact knee. For the deficient too little real world data points are available to create sensible error metrics.

Flexion Angle	RMSE AP (mm)	RMSE IE (deg)
0°	0.34	0.06
30°	0.69	1.35
60°	0.64	1.06
90°	0.39	1.16

Table 3.3: Root Mean Square Error of all discrete knee flexion angles for AP and IE validation for ACL intact.

The root mean square error values indicate a good representation of real world ACL intact knee behaviour. It indicates that it has a maximum average absolute error of 0.69 mm in the AP translation and 1.35 degrees in the IE rotation. These numbers are considered to be sufficient for this research. This also confirms that the model follows the data fit provided.

From this it can be concluded that the model is valid for the application of testing the soft robotic brace in the AP direction and IE direction, because it follows the shape of the motion and the error between the model and real world data is significantly low that it does not affect the essence of the knee behaviour. For this reason, the ACL intact model can also be used to generate the reference motion based on a force and torque profile, more on this in Section 4.3.

For testing the soft robotic brace ACL deficient knee data is used in the model. A comparison will be made between ACL intact model output, ACL deficient model output, and ACL deficient model with soft robotic brace output in Chapter 5 to highlight the performance of the soft robotic brace on recovering healthy motion by reducing laxity.

3.6.2 Validation Surface Fits

To verify the workings surface fit this validation experiment is focused on the interpolated knee flexion angles. This validation experiment also incorporates the extrapolated forces and torques outside the range performed during the real world laxity experiment. Figure 3.16 and 3.17 show AP displacement and IE rotation of the model. These motions have been tested on a larger range of forces and torques to validate the behaviour. This validation is necessary because during gait the forces and torques exceed the values tested in Section 2.6.2. Furthermore, the model has been tested on three additional, interpolated, knee flexion angles.

When looking at all four graphs we can observed the same. The shape of the motion represents realistic behaviour, also at the extrapolated forces and torques. However, the interpolated knee flexion angles result in an unexpected motion. The interpolated results fall between the adjacent knee flexion angles' motion, but not exactly in the middle. E.g. the motion of 45 degrees of knee flexion falls between the motion of 30 and 60 degrees of knee flexion, but not linearly in the middle as expected. This can be derived from the interpolation method used in the surface plot. A linear interpolation is applied on the x-axis, however, to get the motion in the middle this linear interpolation should be applied to the z-axis of the surface plot in Figure 3.10 and 3.11. This is a limitation of MATLAB's `cftool`, and for this research this is accepted.

It can be concluded from that the extrapolated forces and torques are valid up to -400N/400N and -12500Nmm/12500Nmm respectively. The interpolated knee flexion angles are also considered to be valid for the purpose of this research. For this reason the model can be used during continuous knee flexion angle simulations, required for simulating gait.

3.7 Region of Operation

The model allows manipulation of the knee flexion angle in the range of 0 to 90°. It further allows applying forces in the AP direction on the tibia and applying a torque to the tibia in the IE direction. The laxity data, for both ACL intact as ACL deficient, range in knee flexion angle for 0 to 90°, in AP load force from -99.8 N to 99.8 N, and in IE load torque from -5159.7 Nmm to 5159.7 Nmm. In this range the validity of the model can be determined based on goodness of fit as described in Section 3.6.

However, as forces during gait are higher then the above described extremes (OrthoLoad, 2021) the fits are extrapolated in the AP force and IE torque range, as described in Section 3.3.5 and 3.3.6. Because of this extrapolation of the AP and IE data the model theoretically allows for very high forces, practically any force one can imagine. As this model only takes into account the stiffness of AP and IE, with extremely large forces the non-linear behaviour of the tissue surrounding the knee joint will also affect its behaviour. Imagine such a high AP force that the tibia will be ripped off. This is not modelled.

For the above reason the choice has been made to set the operation AP forces from -400N to 400N. The allowed operable IE torque is set from -12500 Nmm to 12500 Nmm. These values have been chosen to allow simulation of gait and are considered result in realistic behaviour in the model, and have been validated in Section 3.6.2. The AP forces during gait vary between -400N posterior and 100N anterior. The extreme IE torques during gait are between -4500Nmm external and 10200Nmm internal (see Section 4.3). These values are within the operable range, allowing for simulation of gait.

3.8 Future Improvements and Shortcomings of the Knee Model

The current model only support AP and IE rotation. A more sophisticated model should also include VV and SI characteristics. Together with dynamic modelling this would result in a more accurate representation of the real world. For this more data must be available, e.g. larger force/torque range, dynamic data, more data points. Currently only 2 data points per knee flexion angle were available for the ACL deficient data set. The maximum force and torque in the available data are 99.8N and 5159.7Nmm, forces during gait are larger. The author is aware that larger forces applied to the knee can be harmful.

Furthermore, the current range of knee flexion angles allow for gait and climbing and descending stairs. However, activities such as getting up out of a chair, riding a bike and squatting require a larger knee flexion angle then 90°. Larger knee flexion angles are not modelled nor can be extrapolated out of the current laxity data.

The ideal force controlled prismatic joint does not represent a soft actuator accurate. To create a more realistic PAM model real world experimental data must be obtained. At the time of writing this experimental data was not available nor were experiments possible due to the COVID-19 pandemic. Interaction of the brace with the leg has also not been modelled. The connection between bones and brace is considered to be rigid in the model.

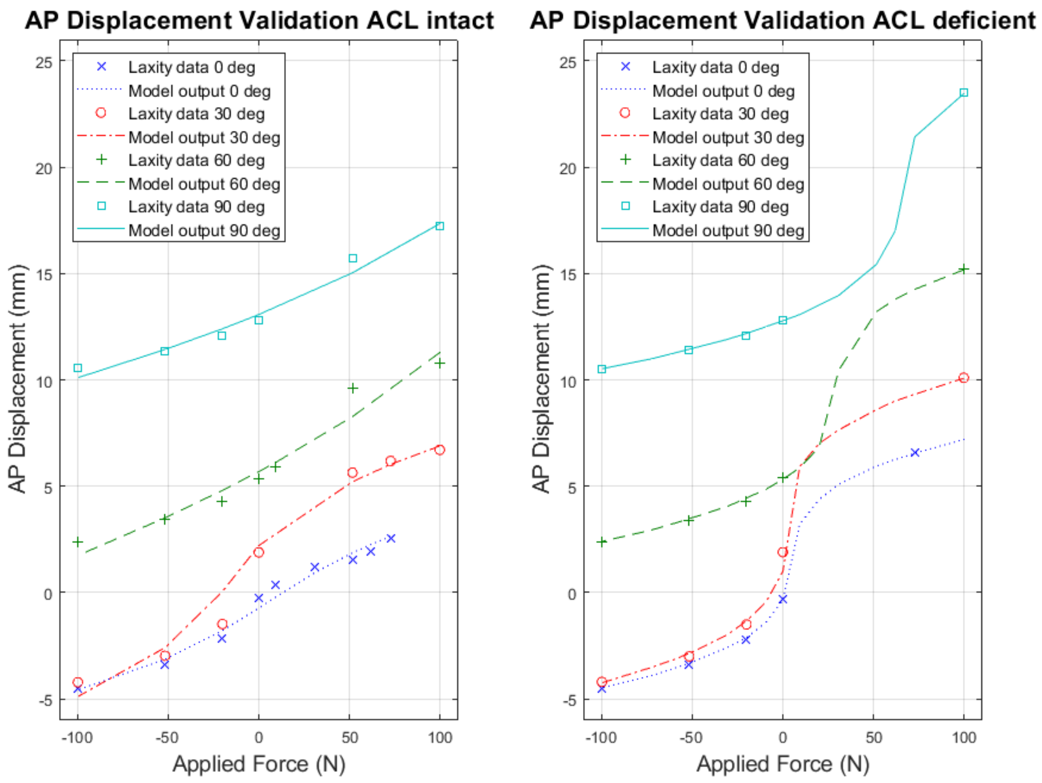

Figure 3.14: AP displacement model validation for ACL intact (left) and ACL deficient (right)

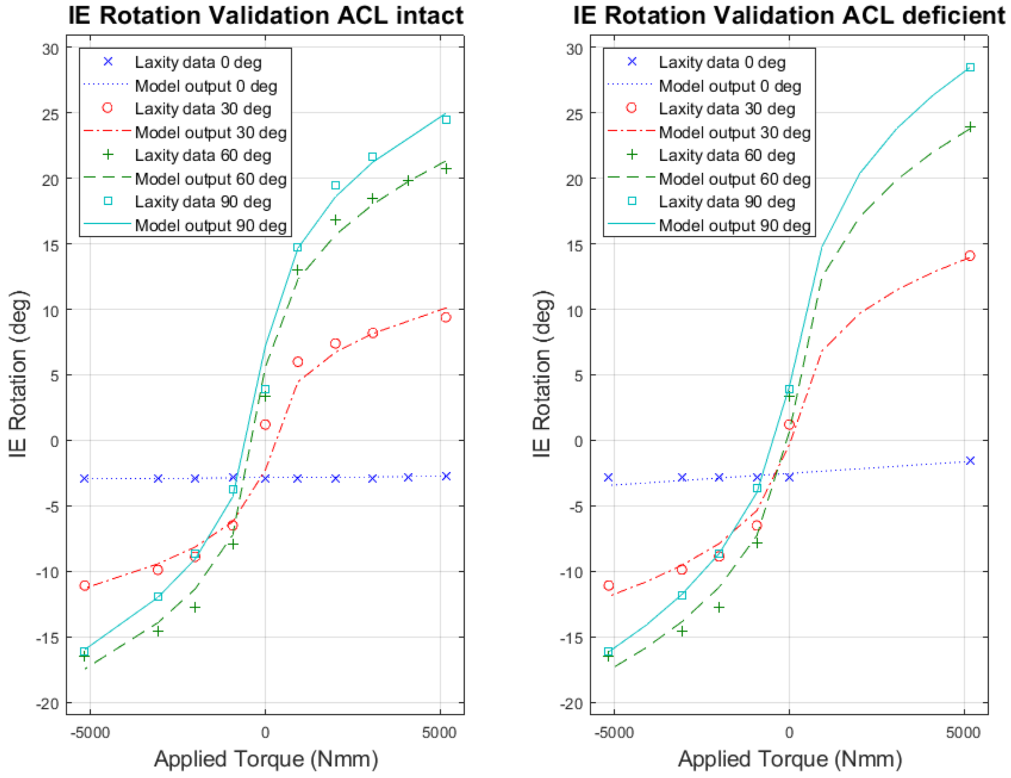

Figure 3.15: IE rotation model validation for ACL intact (left) and ACL deficient (right)

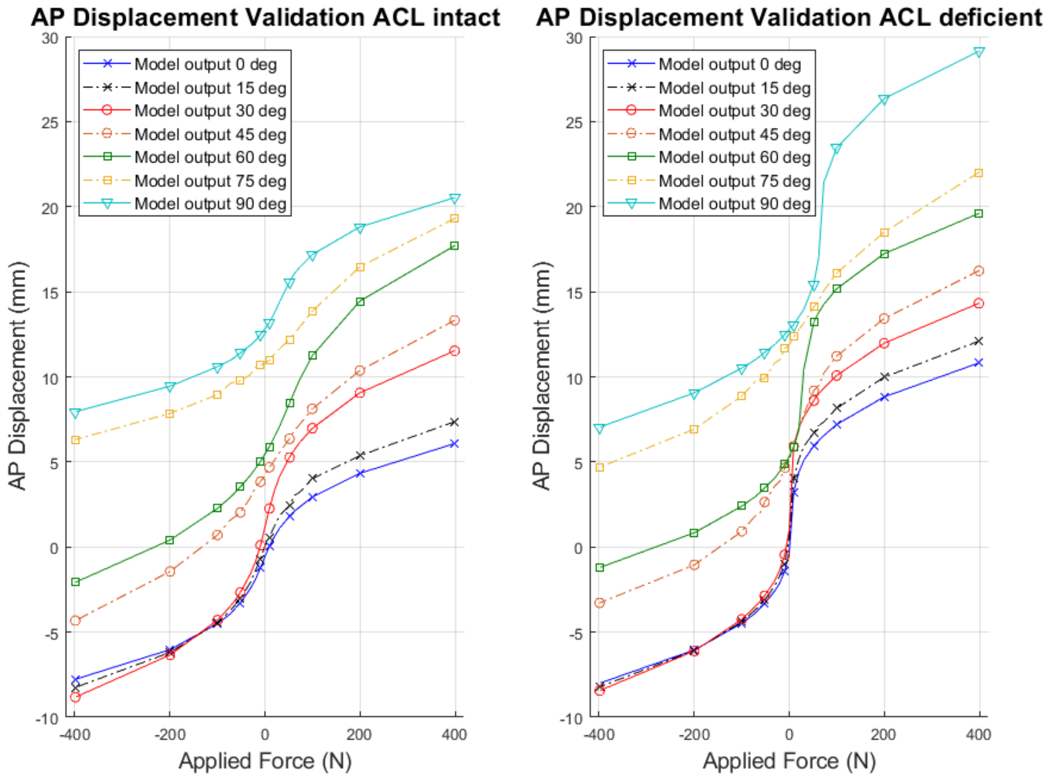

Figure 3.16: AP displacement model validation for ACL intact (left) and ACL deficient (right)

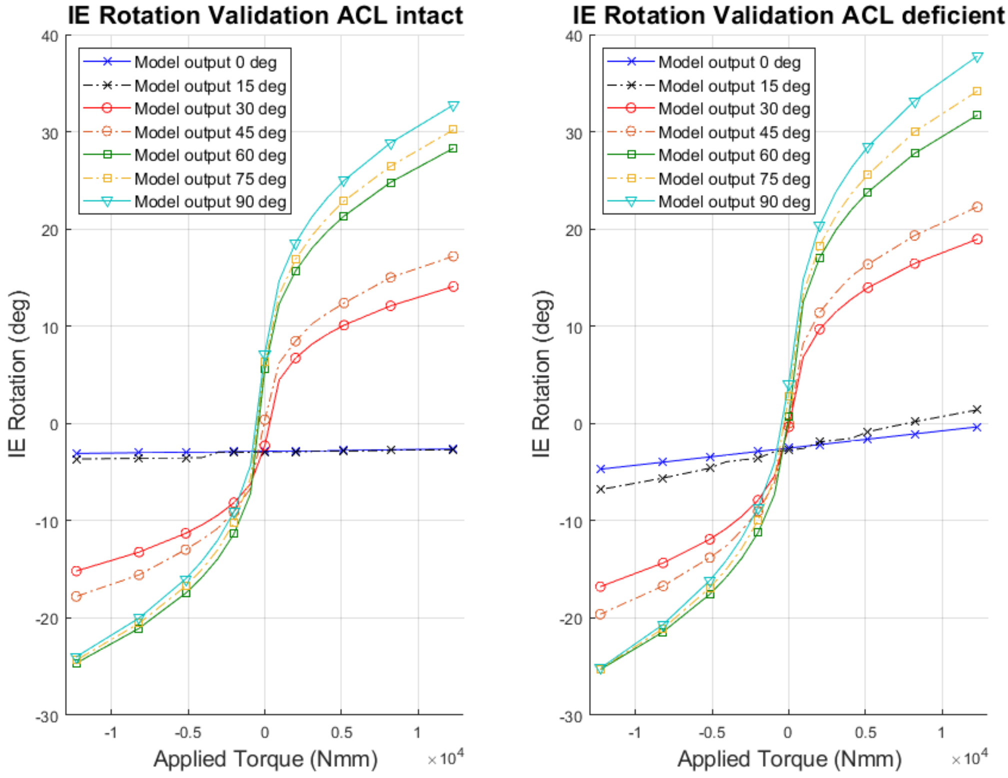

Figure 3.17: IE rotation model validation for ACL intact (left) and ACL deficient (right)

4 Control Design

4.1 Control Introduction

This chapter introduces the control design of this research. To be able to control the soft robotic brace a control system has been specifically build to suit the application of this brace. When designing the control system the following items are considered using the MoSCoW method:

- (Must have) The system must be able to (partly) reduce the AP and IE increased laxity caused by ACL lesion,
- (Must have) The system must be able to be used during gait,
- (Should have) The system should be adaptable to the specific condition of the patient,
- (Would have) The system could be easily usable by clinicians,
- (Will not have) The system will not be able to recover VV laxity caused by ACL lesion.

To elaborate the items named above: the soft robotic brace must be able to reduce the extra motion in AP and IE direction caused by ACL lesion, or in other words must be able to bring the motion back closer to healthy motion of an ACL intact knee. At first, the control system is focused on bringing back this healthy motion during gait. This activity is chosen as it is consider to be one of the most fundamental during daily life. In future research the control can be adapted to different activities.

The third point addresses the adaptability of the brace to the specific condition of the patient is important for optimal recovery and well being of the patient. This would allow faster and more comfortable recovery, and possibly reduce the chances of having a repeating lesion (Chachula et al., 2012). This adaptability can be in the sense of prioritizing certain motion, limiting motion by means of hard limits or by creating a custom reference profile that is optimal for the patient. This can even be a motion profile that incorporates hard limits or gradual stops.

The fourth point mentions the ease-of-use by the clinicians. This is considered to be a non-priority point and will not have a small to no impact when left out. This point is very broad and allows for creative solution. For example in case of only a single affected knee and a healthy knee this could give rise the solution for measuring laxity of both knees. This would then allow the creating of two data sets: a healthy ACL laxity data set and ACL deficient laxity data set. This data can then be used to configure the soft robotic brace to bring back the laxity of the ACL deficient knee to the laxity of the ACL intact knee. This is assuming both knees would have the same healthy laxity before the lesion. For this to be easy for the clinician an easy data capture and data uploading pipeline can be considered.

The last point is obvious as the plant does not include VV capabilities, the control system cannot be used to reduce the laxity in this plane. For this reason this point is in the category will not have. Further research can explore the possibilities of this.

To realise the above, multiple control systems are considered, described in the next section.

4.2 Control Systems Considered

This section highlights different control strategies to use in combination with the plant of Chapter 3. The choice of control strategy is based on the nature of the plant, the characteristics of the actuators used and the nature of the application.

The model has limitations in a way that it does not have real world dynamic data modelled into it. The model is modelled after static data, and as explained in Chapter 3 the model is considered to be of static nature and only validated at slow motion. The lack of dynamic behaviour rules out control strategies such as linear quadratic control (LQR) or other dynamic control strategy such as pole placement strategies. Creation of a dynamic model could be realized by using obtaining the dynamics from literature or using the OpenSim, a open source software system for biomechanical modeling. However, combining the dynamic properties with the static data from Naghibi Beidokhti et al. (2017a, 2020) would be hard to validate.

As there is a model of the plant available, one could consider using model predictive control (MPC). This method is usually used for very complex dynamical systems, but can be used for simpler systems as well. MPC uses the current state of the plant, the model of the plant and the reference signal to calculate future outputs of the plant. This is done for each time-step and creates a control signal for $[t, t+p]$. It will then deploy only the first step of the computed control signal at time t. After, the system repeats the computations based on the new state. This method is very compute-intensive as it computes the plant outputs multiple time-steps in advance every time-step of the system.

Another important aspect for choosing a control strategy is the type of actuators. In the soft robotic knee brace pneumatic artificial actuators are used. These type of actuators are considered to be slow compared to traditional rigid actuators, as described in Section 2.5. A control strategy that uses direct feedback methods would result in instability. Together with the lack of dynamic characterisation of the model control strategies that focus on rapid error convergence can be neglected, as the model lacks dynamic behaviour and the actuators used form a hardware limitation.

Last, the nature of the application could influence the choice of control strategy. In the case of the soft robotic knee brace, during gait the motion of the application is very repetitive. For this case the control method Iterative Learning Controller (ILC) can be considered. By observing the error of each cycle, the command for the next cycle can be improved. This aims for a error converging to zero. The work of Obbink (2019) demonstrates the use of ILC in repetitive motion of a liver phantom using soft actuators. Another option for repetitive applications is the use of adaptive control, in which instead of changing the control signal as done in ILC control the parameters of the controller are changed to obtain zero error.

Another control method that has recently been emerging are in the group of the so-called intelligent control techniques. Artificial neural networks (ANN) is one of those. These type of controllers all have in common that they require some sort of learning. This must not be confused with iterative learning controller (ILC), as ILC includes a very specific type of learning based on the error of the input and output and learns on-the-fly (Moore, 1993).

Table 4.1 gives a summary of the control strategies that are considered specifically for the scenario of the soft robotic knee brace. From the description and table it can be concluded that there are two candidates for a control law: ILC and adaptive control. As described above, the main difference between the two is that ILC changes the control signal while adaptive control changes the control parameters. While ILC works in cycles, and technically not in real-time it can be seen as an open-loop control law. Adaptive control composes the control signal by processing the error in real-time, thus can be seen as a closed-loop control law and still needing a proper controller. For this reason ILC is chosen for the reason it is less complex.

	DC	MPC	DFB	ILC	AC	IC
Error Convergence	++	++	+	-	-	-/+
Complexity	-/+	-	++	++	+	- -
Computational intensive	-/+	- -	++	-/+	-/+	- -
Robustness	++	+	-	++	++	-/+
Adaptability	- -	- -	- -	++	++	-/+

Table 4.1: Control strategies considered. '+' is positive, '-' is negative. DC: Dynamic Control (LQR, pole placement), MPC: Model Predictive Control, DFB: Direct Feedback (simple PID), ILC: Iterative Learning Controller, AC: Adaptive Control, IC: Intelligent Control

By choosing ILC as the control law it is allows to operate the control system in a open-loop after learning. This can be useful in a case where the brace is in learning mode under supervision of the clinician with inertial sensors attached to the upper- and lower leg. After learning the sensors can be removed, assumed the motion stays the same. The ILC is further explained in Section 4.4.2.

As described above an ILC modifies the control signal each iteration. The first iteration there has not been any error[1] recorded, for this a inverse plant controller is used that will deliver an approximate initial control signal to the ILC. This increases the convergence rate by already giving an approximate control signal. However, the use of the inverse plant controller needs to measure the state of the plant creating a closed-loop like system. It is not self-correcting like a proper closed-loop system. More on this in Section 4.4.1.

To combat the non-linear behaviour of the actuators, this referring to the saturation of the output force, a state machine controller has been made that accepts the individual force control signal components for AP displacement and IE rotation. Using these signals it will compute the output signal for each individual PAM. The controller can prioritise AP or IE if one, or both, of the actuators saturate, more on this in Section 4.5.1.

Figure 4.1 shows an overview in which the controller is implemented into the full system including the reference signal, plant and output. For the system it is assumed to have inertial sensors that are able to measure the tibial AP displacement and IE rotation. It is also assumed that there is a sensor in the brace measuring the flexion and extension of the knee joint.

4.3 Reference Signal

The reference signals provided to the controller, as shown in Figure 4.1 in the green area, represents ACL intact motion, specifically AP displacement and IE rotation. The goal for the controller is then to by means of actuating the PAMs to reach this motion, or get to an optimum that is physically possible. The reference signal for the controller is motion, in millimeters for AP displacement and degrees for IE rotation. This motion is in-sync with the knee flexion angle

[1] The error is expressed in the difference between the motion of the ACL deficient knee with soft robotic knee brace and the ACL intact knee motion

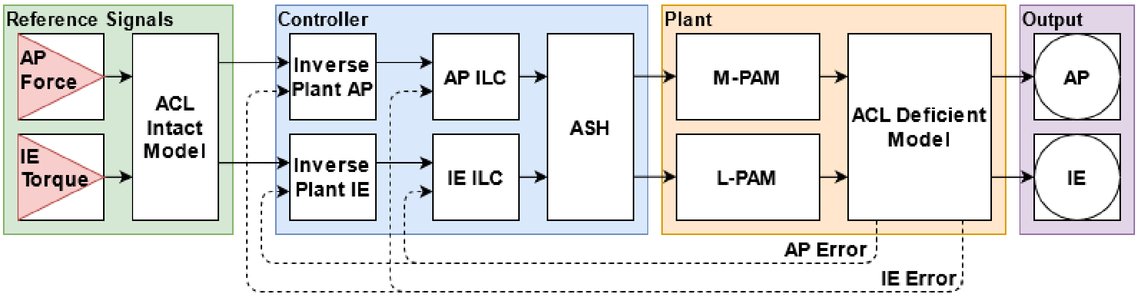

Figure 4.1: Total overview of the plant, controller and connections

signal described in Section 3.3.2.

The ACL intact motion is obtained by uploading a force and torque profile, for e.g. gait, to the knee model described in Section 3.6 with ACL intact characteristics. This force and torque profile is applied to that model as external force as described in Section 3.5. The ACL intact knee model will the compute corresponding AP and IE motion. According to the validation results in Section 3.6 the output motion can be used as reference signal.

This method of generating the reference signal is chosen as it can give a direct comparison between the models; ACL intact knee, ACL deficient knee, and ACL deficient knee with soft robotic brace. Creating another fit from force and torque to AP displacement and IE rotation directly, from the ACL intact data in Section 2.6, will introduce discrepancies in the comparison between the models. This method will be more computing intensive, but will result is a more accurate result for this research. A schematic is shown in Figure 4.2.

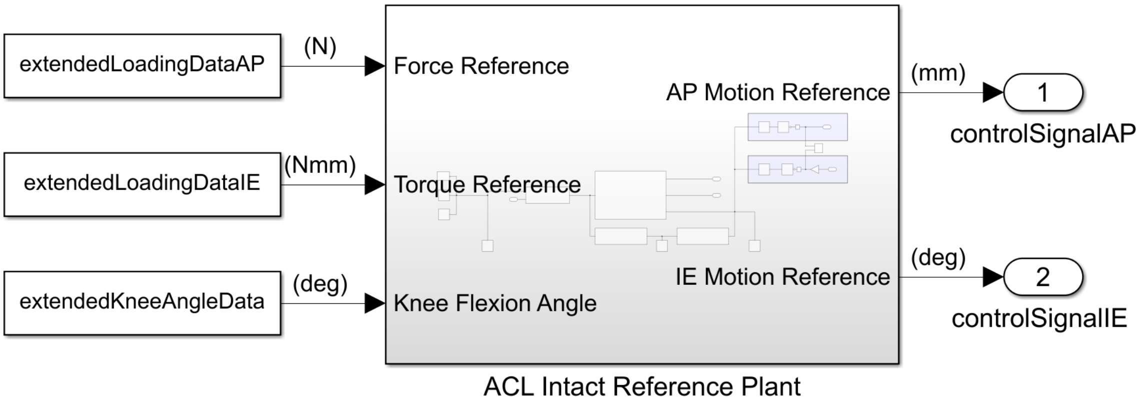

Figure 4.2: Gait force and torque to reference motion

The figure shows the reference plant with ACL intact characteristics. The plant is supplied with gait AP force, in Newtons, and IE torque, in Newton millimeter, and the knee flexion angle, in degrees. The force and torque data is supplied by OrthoLoad (2021). Specifically, the knee joint data of patient data K1L[2] walking at 4km/h on a treadmill wearing basic sports shoes. The data is supplied in percentage of body weight (%BW). This data is then applied on the body weight of the patient data from which the laxity data is obtained, as of Section 2.6.2.[3] Taken

[2] Database filename: k1l_180908_1_15p

[3] Cardavic knee identifier: C926B_R. BW: 87.5 kg

into account the data from OrthoLoad (2021) is for a left knee and the data from Section 2.6.2 is for a right knee. The directions are adapted to the left knee modelled in this research.

Figure 4.3 shows to AP force, IE torque and knee flexion angle of the above mentioned gait of 12 seconds corrected for the body weight of patient C926B_R.

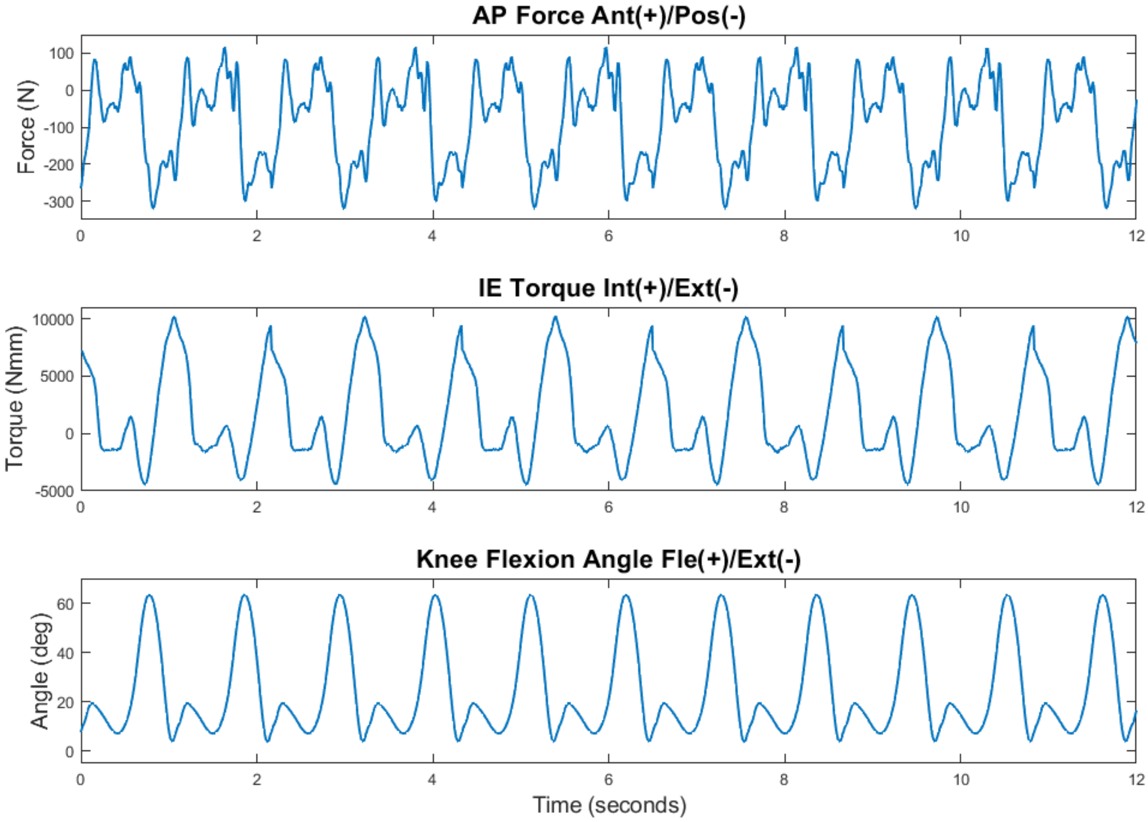

Figure 4.3: Left knee gait AP force, IE torque and knee flexion angle

4.4 Controller Overview

As described in Section 4.2 the control system can be subdivided in three stages: a plant inverse controller, an iterative learning controller (ILC), and a state machine actuator saturation handler. These three stages can be seen in the blue controller area in Figure 4.1. Figure 4.4 shows the Simulink implementation of these three stages.

The workings of the controller as a whole is to compare a reference signal, which is ACL intact knee motion, to the motion of the knee on which the soft robotic knee brace is deployed, a ACL deficient knee. This is not a self-correcting direct feedback used in feedback systems, the error is recorded in iterations by the ILC. The controller will then reduce the motion error signal by means of actuating the PAMs which are located on both sides of the soft robotic knee brace. The goal is to converge to zero error which would results in reducing the excess laxity back to laxity compared to a ACL intact knee.

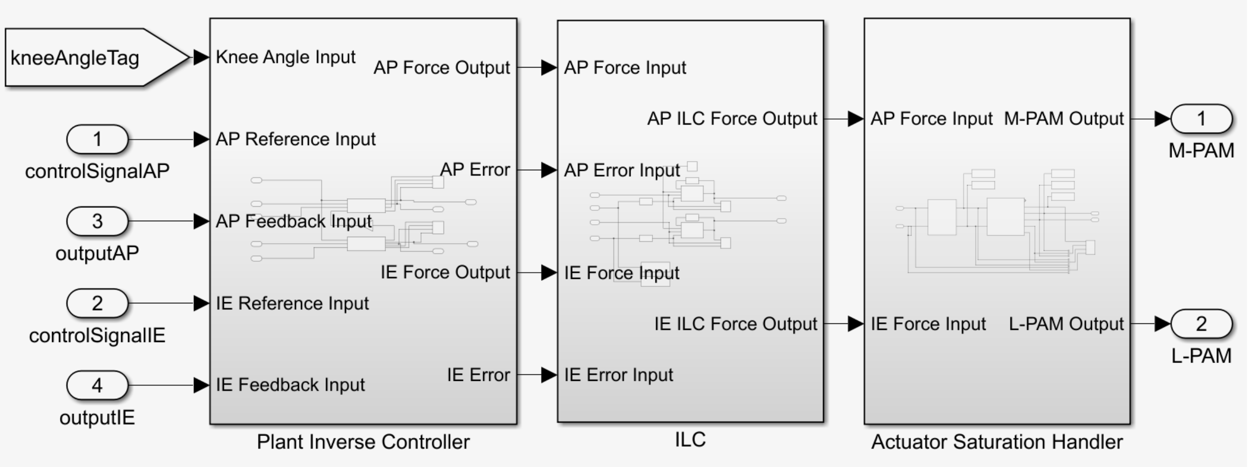

Figure 4.4: Overview of the controller in Simulink

4.4.1 Inverse Plant Controller

The first stage of the control system is the inverse plant controller. This controller combines inverse plant modelling in a feedforward type controller and feedback to determine the current state of the plant, with the states being the AP displacement and the IE rotation. The AP displacement and IE rotation both have their own inverse plant controller as seen in Figure 4.1.

The inverse plant is obtained by supplying the plant with actuator forces and measuring the output motion. This actuator force is delivered by means of AP and IE actuator force components. An AP force component is delivered to both actuators. This is done in the range of [0N 80N], in other words from no actuation (0N) to maximum actuation (80N). The IE force component is delivered as a difference between the medial and lateral actuator. This is done in the range of [-80N 80N]. For clarification, -80N means that the lateral actuator is 80N above the medial actuator inducing an external torque. For the plant inverse mapping the medial actuator is set to 0N. The same goes for 80N but then the medial actuator is set at 80N, inducing a internal torque. The input AP and IE force components are then stored together with their resulting motion. A curve or surface fit has been created and resulting in a static plant inverse mapping. The MATLAB code for the creation of these fits is shown in appendix J.

The controller receives five inputs: the reference signal for AP displacement and IE rotation, as described in Section 4.3, the state feedback signal measuring the AP and IE state of the plant, and the current flexion angle of the knee. The two reference signals both pass through the inverse of the plant, yielding two actuator forces components. These are the actuator forces components needed to reach the reference state if there were no external forces acting on the model. However, during gait there are non-zero external forces acting on the plant. For this reason the current state of the plant is measured by means of feedback.

The feedback signal is also passed through the same inverse of the plant to obtain which, AP and IE, force component is already acting on the plant. Comparing both outputs of the inverse plant of the reference input and of the state feedback input will yield the AP and IE actuator force components needed to get the output to the reference signal. Equation 4.1 and 4.2 show a simplified mathematical description of the process for AP and IE respectively.

$$F_{AP} = f(x_r) - f(x_{AP}) \tag{4.1}$$

With F_{AP} the resulting AP actuator force component (*AP Force Output* of the Plant Inverse Controller in Figure 4.4), $f(x_r)$ the AP actuator force needed for the reference state x_r (*AP Reference Input* in Figure 4.4), and $f(x_{AP})$ the AP actuator force component already present at the current state x_{AP} (*AP Feedback Input* in Figure 4.4). Notice that both components use the same AP inverse plant model function $f(x)$.

$$F_{IE} = g(\theta_r) - g(\theta_{IE}) \tag{4.2}$$

With F_{IE} the resulting IE actuator force component (IE Force Output of the Plant Inverse Controller in Figure 4.4), $g(\theta_r)$ and $g(\theta_{IE})$ the IE actuator forces of the reference state θ_r and current state θ_{IE} (*IE reference Input* and *IE Feedback Input* in Figure 4.4). Both use the same IE inverse plant $g(\theta)$.

One peculiar aspect that might confuse the attentive reader is that for IE rotation an output force F_{IE} is computed, instead of a torque as expected with rotation. However, these are actuator forces components. A difference in medial and lateral actuator forces will induce a torque, and resulting rotation. This methodology uses actuator forces that are positive to be a resulting internal rotation and negative to be resulting external rotation. As expected, an F_{IE} equal to zero will induce no torque. The inverse model is build on model having input actuator forces, medial and lateral, and output AP displacement and IE rotation. Using this method the actuator force components for AP and IE motion can easily be added together to obtain the actual actuator force. This technique is used in the actuator saturation handler in Section 4.5.1.

These two forces, AP and IE actuator force components, are then passed through together with the AP state error, in mm, and IE state error, in degrees, to the ILC (see Figure 4.4), which is described next.

4.4.2 Iterative Learning Controller

Because the motion, human gait, is a repetitive motion, an iterative learning controller is able to learn from each iteration. As each iteration is assumed to be the same, the controller will see the same behaviour and thus the same error each time. The ILC will take this past error and adjust the control signal to the plant for future iterations to try to obtain a zero error.

A basic ILC has the form:

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1) \tag{4.3}$$

With $u_{k+1}(t)$ the control signal for the next iteration at time t, $u_k(t)$ the current control signal at time t, γ the learning gain, and $e_k(t+1)$ the past error between the actual output y_k and desired output y_d. This is also known as the 'Arimoto' algorithm. A schematic is shown in Figure 4.5.

The system has two individual ILC's, one for AP and one for IE, as depicted in Figure 4.1. They work on the AP actuator force component and IE actuator force component respectively. The ILC's are placed behind the inverse plant controller. This way u_k does not start at zero, but already has some non-zero value. This means that the ILC has to compensate less error, which results in a quicker error convergence.

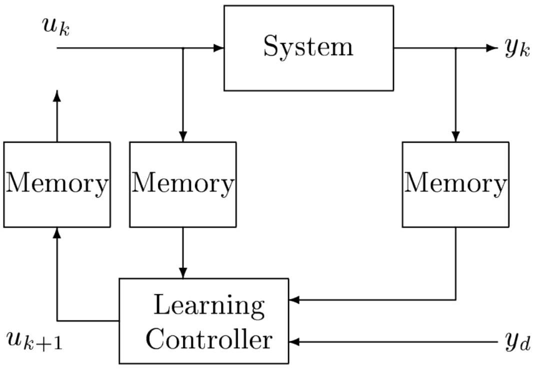

Figure 4.5: ILC schematic (Moore, 1993)

4.5 PAM Control

4.5.1 Actuator Saturation Handler

The AP and IE forces from the ILC are passed through the actuator saturation handler, shown as *ASH* in Figure 4.1. This stage consists of two main parts, shown in Figure 4.6. First part is converting the AP and IE forces to medial and lateral actuator forces, F^*_{MPAM} and F^*_{LPAM} respectively. This is done according to Equation 4.4 and 4.5. the * indicates that these are pre-saturation forces, meaning that they can realise healthy AP and IE motion and do not take into account the operating range of the PAMs. Most of the time these computed pre-saturation forces combined give actuator forces outside the operating range of the PAMs. The operating range, minimum and maximum force, is depended on the hardware of the PAMs.

$$F^*_{MPAM} = F^*_{AP} + \frac{F^*_{IE}}{2} \tag{4.4}$$

$$F^*_{LPAM} = F^*_{AP} - \frac{F^*_{IE}}{2} \tag{4.5}$$

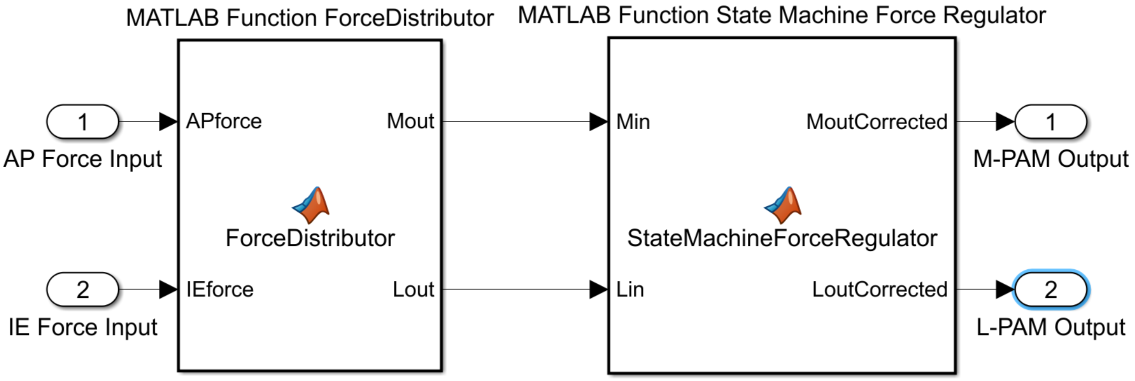

Figure 4.6: Actuator Saturation Handler in Simulink

This addition is possible because of the method in which the AP and IE forces are used throughout the controller. This can best be explained as: AP forces are induced by setting the same actuator force on both actuators (directly linked to the average force), IE torques are induced by setting a difference between the actuator forces (directly linked to the difference in force). By adding and subtracting the same amount the AP displacement, average force, stays the same. The assumption made here is that the rotational axis of the tibia is in the centre of the tibial brace support plate. Example: a larger F_{MPAM} compared to F_{LPAM} will introduce an internal torque, and vice versa will introduce an external torque.

Once the AP and IE force components are converted into medial and lateral actuator forces, the actuator forces are passed to the seconds part of the actuator saturation handler: a state machine. The main purpose of this state machine is to adapt the medial and lateral actuator forces in case if one, or both, saturate.

The state machine can, in case if one actuator saturates, keep the other actuator at a output force that either the average or the difference between both actuator forces stays the same, or a weighted combination of the average and difference. Thus prioritizing AP or IE, or a weighted value between them. This weight is called the *AP-IE prioritization weight r*. If $r = 1$ (full AP-prioritization) the state machine will keep the average of the actuator forces the same as the average of the optimal actuator forces F^*_{MPAM} and F^*_{LPAM} in Equation 4.4 and 4.5. If it is not possible to keep this average, e.g. if the average is above or below the saturation limit of the actuators, the state machine will saturate the output forces to the nearest actuator limits.

If $r = 0$ (full IE-prioritization) the state machine will keep the difference between the actuator forces the same as the difference of the optimal actuator forces F^*_{MPAM} and F^*_{LPAM}. Again, if this is not possible the state machine will saturate the actuator forces to obtain the closest difference to the optimal difference.

Figure 4.7 shows a example visualisation of both a medial and lateral PAM force signal. It furthermore shows the input and output average and differences of these two PAM forces. Three time frames (I, II, and III) have been highlighted that show behaviour of the actuator saturation handler:

- I: Both the input and output average, thus AP, and difference, thus IE, can be preserved as both PAMs are in their operating range of 0N to 80N. This can be seen as the input and output average and difference are the same.
- II: The average PAM input force is below 0N, so both PAMs saturate at 0N to get the best possible AP outcome as the system is in AP-prioritization. This can be seen as the input and output average (AP) are not equal. The same is the case of the input and output difference (IE).
- III: The medial PAM saturates. As the control sample is in AP-prioritization, the actuator saturation handler will prioritize AP, and thus the average input force. This is done by solely actuating the lateral PAM to keep input and output average force equal. It can be seen that the input and output average force stay equal, but at a cost of IE torque, as the input and output difference is not equal.

The AP-IE prioritization weight can be set to any value between 0 and 1 and the state machine will weight the average optimal actuator force and difference between the optimal actuator forces linearly. If the actuator forces fall within the saturation limits of the PAM actuators the state machine will pass these through to the output of the actuator saturation handler unaltered. All individual states can be found in the MATLAB code in appendix E.

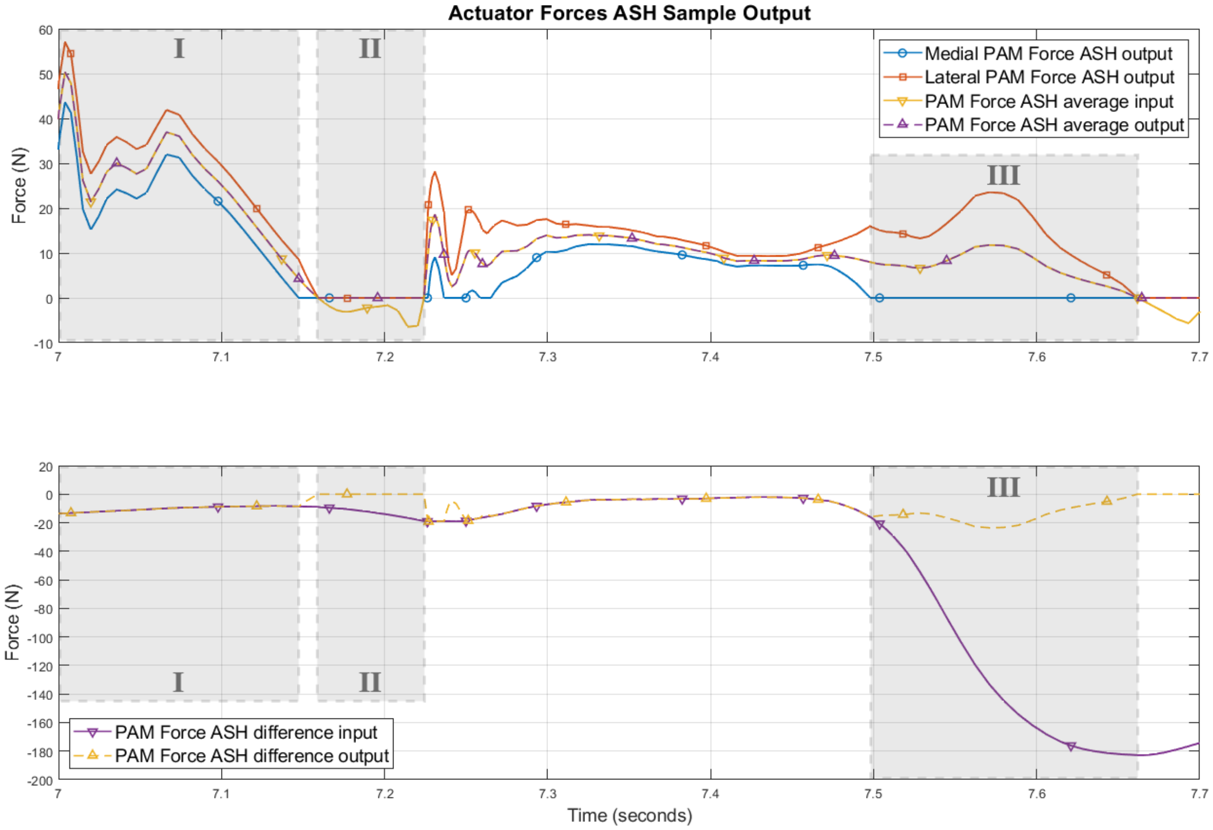

Figure 4.7: Sample control signal output of the Actuator Saturation Handler in AP-priority mode $r = 1$. The individual PAM forces and averages have been separated from the differences to make the graph more clear.

4.5.2 Force to Pressure

After the medial and lateral PAM actuator forces are computed by the controller, that meaning all three stages, these actuator forces can be converted to air pressure. This air pressure can then be applied to the PAMs.

For this research the mapping between PAM output force and PAM pressure is assumed to be linear and the conversion from input to output to be ideal. It is further be assumed to have a PID controller in the PAM regulator that will match the output force to the specified input reference force. The addition of a rate limiter has been added to introduce slowness of the PAMs. The exact slowness is not known, but will be experimented with.

5 Results & Discussion

In this chapter the results of this research are presented and discussed. The control system from Chapter 4 has been placed in an open loop with the plant from Chapter 3 as depicted in Figure 4.1. The inverse plant controller measures the state of the system depicted by the dotted connections. This feedback is also used by the ILC. There are used to store the difference[1] of each iteration in memory, thus are not directly in the loop.

First, the individual stages of the controller are evaluated at discrete knee flexion angles. This is done to omit errors of interpolation between the four discrete knee flexion angles and to isolate the performance of each individual stage. The goal of these experiments is thus also to validate their purpose. The ILC is evaluated first in Section 5.2 at 30 degrees of knee flexion in AP-prioritization. These results give us an impression how the ILC behaves and if it works as intended.

Once the ILC is evaluated in AP-prioritization, the Actuator Saturation Handler's performance is evaluated in Section 5.3. This is done by evaluating the same experiments in IE-prioritization and weighting between AP and IE and measuring the performance and comparing them with AP-prioritization. Then in Section 5.4 the effect of the inverse plant controller is measured and compared to the case without the inverse plant controller.

While the above experiments are performed with ideal actuators, in Section 5.5 the same experiment in discrete knee flexion is done with various actuator speeds of the PAMs to measure the robustness of the system.

Once the controller stages are evaluated at a discrete knee flexion angle, the concluded controller is testing during gait with a continuous changing knee flexion angle in Section 5.6.

During all experiments the same gait forces and torques are applied to the model and reference model as described in Section 4.3, disregard if the simulation is on a discrete or continuous knee flexion angle.

5.1 Motion Recovery

During all experiments the motion recovery of the ACL deficient knee model with soft robotic knee brace is measured. This recovery is based on the difference in motion between the ACL deficient knee model with soft robotic knee brace and the ACL intact knee model motion. This difference is then divided by the difference in motion between the ACL deficient knee model without soft robotic knee brace and the ACL intact knee model motion. This way a percentage recovery is obtained.

The recovery and normalized recovery are shown in Equation 5.1

$$Recovery(\%) = 100\% - \frac{MAE\left(x_{brace}(t),\ x_{intact}(t)\right)}{MAE\left(x_{deficient}(t),\ x_{intact}(t)\right)} \cdot 100\% \tag{5.1}$$

[1] When speaking of the difference this indicates the difference between the motion of an ACL deficient knee with soft robotic brace and an ACL intact knee.

With MAE the Mean Average Error operation, $x_{brace}(t)$ the motion of the ACL deficient knee model with soft robotic knee brace, $x_{deficient}(t)$ the motion of the ACL deficient knee model without soft robotic knee brace, and $x_{intact}(t)$ the motion of the ACL intact knee model.

5.2 ILC Performance

To evaluate the ILC performance it is chosen to put the Actuator Saturation Handler in AP-prioritization. This choice between full AP-prioritization or IE-prioritization is made arbitrary. Full IE-prioritization is later shown in Section 5.3. To highlight the performance of the ILC only, the inverse plant controller has been disabled. The inverse plant controller is later shown in Section 5.4.

The input reference signal is provided according to Section 4.3. For the discrete case this same reference signal is used but at non-changing knee flexion angles.

Figure 5.1 shows the AP and IE output of the plant at 30 degrees of discrete knee flexion. For comparison the same AP and IE output have been plotted for an ACL intact knee and an ACL deficient knee without soft robotic brace. Figure 5.2 shows the actuator force of the same simulation of the knee in 30 degrees of discrete knee flexion.

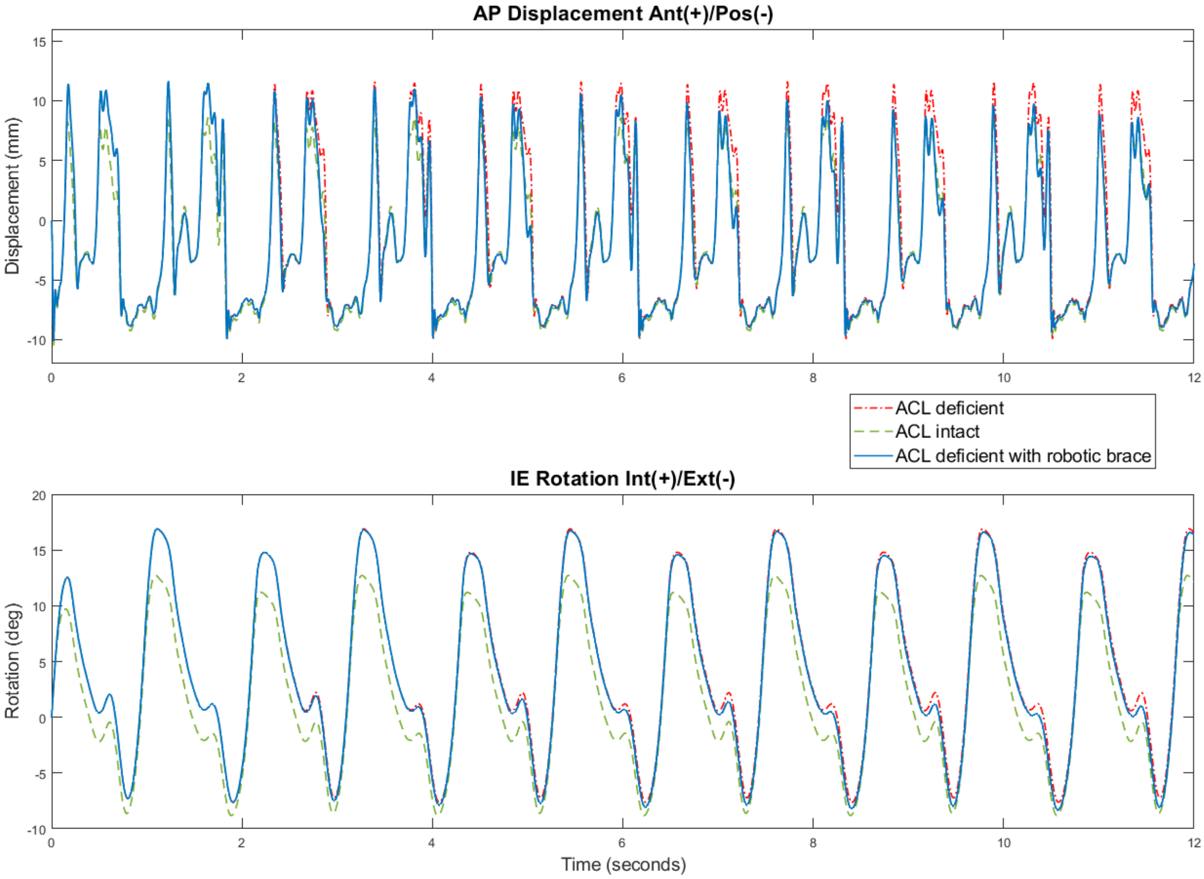

Figure 5.1: ILC results at 30 degrees of discrete knee flexion. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

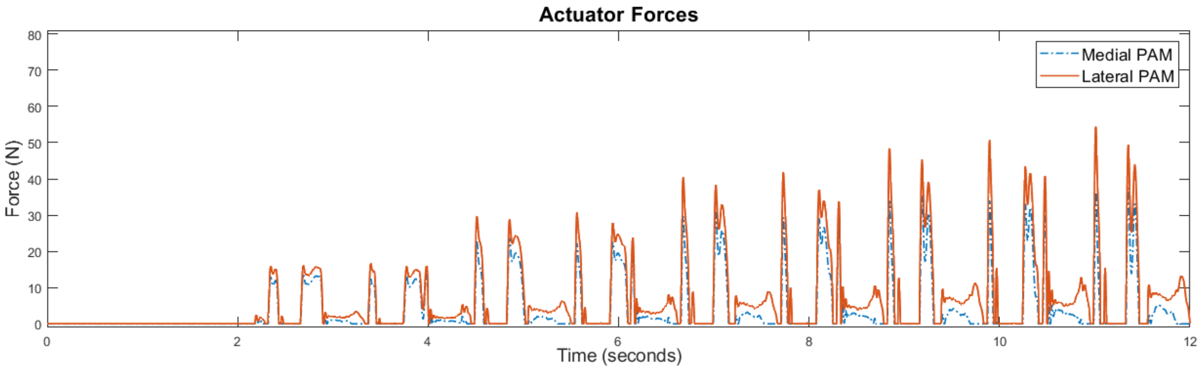

Figure 5.2: Actuator forces for the motion result in Figure 5.1

It can be seen by visually inspecting both the motion and the actuator forces that the ILC learns when time progresses. During the first two seconds the motion of the ACL deficient knee with soft robotic brace has the same exact motion as the ACL deficient knee without the brace. This can also concluded when looking at the actuator forces during the first two seconds. After some time it can be seen that the AP motion of the ACL deficient knee with soft robotic brace converges to the AP motion of the ACL intact knee. Resulting in a recovery of 68.3% in the AP displacement and 24.6% in the IE rotation during the last two seconds of the simulations.

It can also be observed that the IE rotation converges to the ACL intact reference rotation, however in a lesser magnitude than the AP displacement. The reasoning for this can be derived from the actuator forces in Figure 5.2. At the times that the IE rotation is most internal, approximately between 10° and 17° internal, one of the two actuators are saturated at 0N. These are also the times where the IE difference is the biggest. This saturation leaves the controller with no headroom to realize both AP and IE fully, thus relying on the prioritization mode set. In other words: the difference between both actuator forces (linked to IE rotation) cannot be increased while keeping the average of both actuator forces (linked to AP displacement) the same, as described in Section 4.5.1. Section 5.3 will show the same experiment, but in IE-prioritization.

However, there is more headroom left to realize more AP displacement as there is more room to increase this average force. This can be seen from the actuator force graph. The maximum force used in 60N, by a single PAM, while the maximum force is 80N.

The same experiment has been repeated for different values of the learning gain γ_{AP} and γ_{IE}, for the AP ILC module and IE ILC module respectively. Figure 5.3 shows the recovery of AP and IE motion for different values of γ_{AP} and γ_{IE}. For the influence of γ_{AP} on the AP recovery the system is placed in AP-prioritization ($r = 1$). Respectively for γ_{IE} in IE-prioritization ($r = 0$).

However, just looking at the recovery percentages alone is not sufficient. When looking at the graphs in appendix C it can be seen that values of γ_{AP} of 1 and 2 the difference does not converge that well, and for values of 6 and 8 it overshoots and is too aggressive. This can also been seen in the recovery numbers in Figure 5.3. For values of γ_{IE} larger then 2 the motion behaves oscillatory, becoming unstable. Especially seen at values of 6 and 8.

For the above reason the values of $\gamma_{AP} = 4$ and $\gamma_{IE} = 1$. The latter is chosen to prevent any dangerous while still preserve converging behaviour.

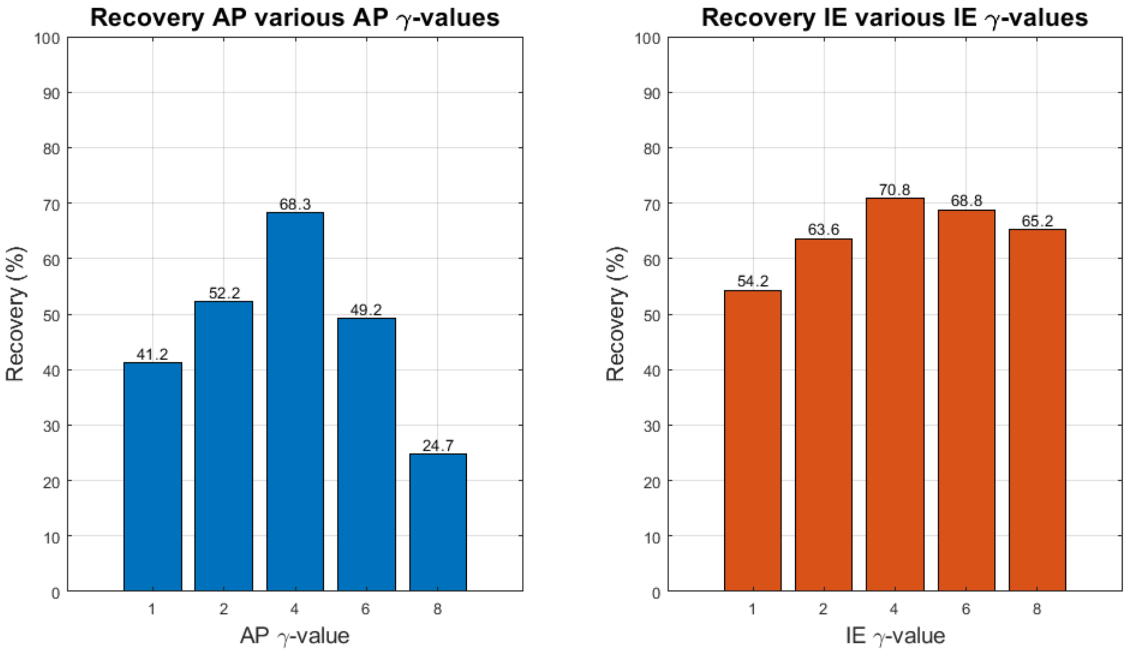

Figure 5.3: Recovery percentages at 30 degrees of discrete knee flexion for different values of γ_{AP} (left) and γ_{IE} (right)

5.3 Actuator Saturation Handler Performance

To evaluate the performance of the actuator saturation handler multiple experiments are conducted. The AP-IE prioritisation weight is varied between 0 and 1. This will show us how the system will handle different prioritizations between AP and IE motion.

Figure 5.1 and 5.2 already show the actuator saturation handle in full AP-prioritization ($r = 1$). Observing the motion together with the actuator forces it can be seen that the AP displacement is prioritized over the IE rotation. While time progresses the AP difference converges while the IE difference does not with the same magnitude. Looking at the actuator forces in Figure 5.2 shows us that the system will compute actuator forces that will realize the correct average force between the two actuators. This results in an effective AP force in the posterior direction. However, during these periods it can also be seen that one actuator saturates at 0N. This way there is no more headroom to induce a torque by the two actuators. This physical limitation limits the compensation of IE rotation in AP-prioritization mode.

Figure 5.4 and 5.5 show the exact same experiment (30 degrees of discrete knee flexion with gait force and torque), but with full IE-prioritization ($r = 0$).

As seen in the figure the IE motion converges towards the ACL intact motion. At the large peaks the motion converges slow, this can be traced back to the external torques applied as they are at their largest during these peaks. However, Figure 5.5 shows that there is space to satisfy a larger recovery. This can be accelerated by setting a more aggressive IE learning rate in the ILC as explained in the Section 5.2. In the range of 5° internal to 10° external rotation the most visible recovery is made. During these times little external torque is applied, so the soft robotic knee brace can easily recovery the motion. In IE-prioritization an AP displacement recovery of 52.3% and IE rotation recovery of 54.2% is made.

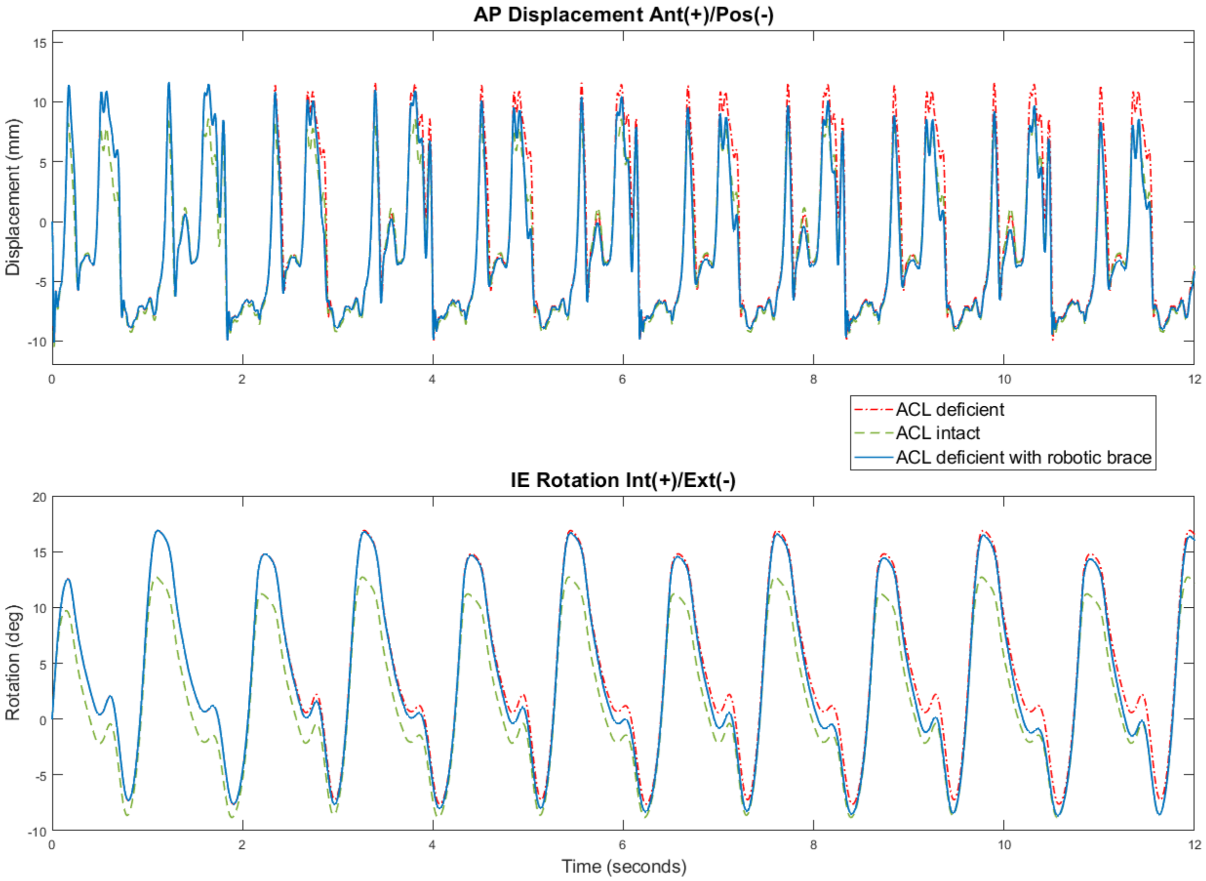

Figure 5.4: ILC results at 30 degrees of discrete knee flexion in IE-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

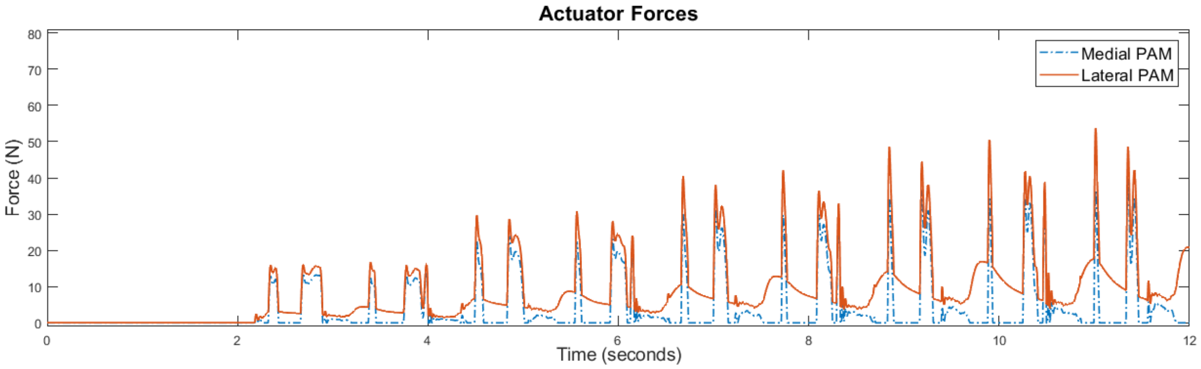

Figure 5.5: Actuator forces for the motion result in Figure 5.4

Visually comparing Figure 5.4 and to the AP-prioritization case in Figure 5.1 it can be seen the IE rotation converges more from the ACL deficient knee towards the ACL intact knee, especially in the regions of around 0° of rotation. However, it can be seen that at the higher internal rotations, around 10° to 17°, the recovery is minimal although higher then in the AP-prioritization case. A longer simulation time would visualize the recovery more clearly as the ILC will have more iterations to converge.

Looking at the AP displacement during IE-prioritization it can be observed that the motion of the ACL deficient knee with soft robotic brace converges towards the ACL intact motion. However, IE rotation has not converged fully towards the ACL intact motion. Meaning the ILC has needs more iterations to learn, seen by the still increasing actuation forces and leftover room in the operating range. If that would be the case, AP displacement might suffer during IE-prioritization. Here, a longer simulation time would visualize this effect. This effect can also be highlighted if the inverse plant controller gives the ILC an initial control signal, shown in Section 5.4 for the IE-prioritization case.

The same experiments in discrete knee flexion angle as above have been performed for the AP-IE prioritization values r in between 1 and 0. Figure 5.6 shows us different values of r with the resulting recovery in AP and IE of the last 2 seconds of the simulations. For the values of $r = 1$ and $r = 0$ the motion can be seen in Figure 5.1 and 5.4 respectively. The results clearly show what is expected: a bias towards AP prioritization will increase the recovery in the AP displacement, and vice versa for IE prioritization.

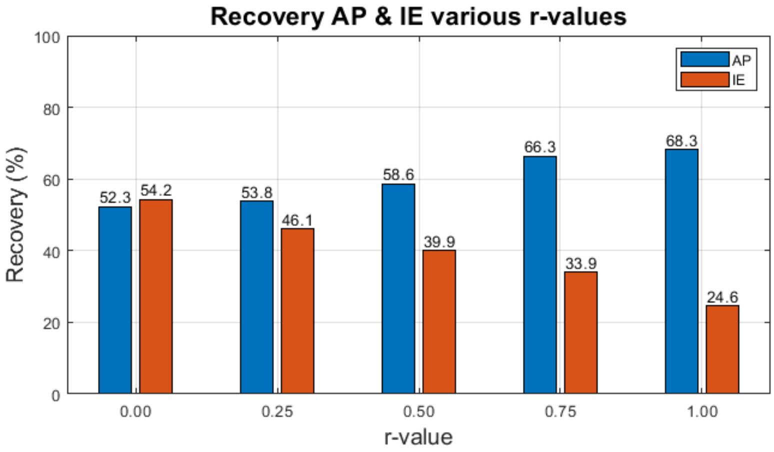

Figure 5.6: Recovery percentages for different AP-IE prioritization values r at 30 degrees of discrete knee flexion. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

5.4 Inverse Plant Controller Performance

To evaluate the inverse plant controller the same experiment as in Section 5.2 (30 degrees of discrete knee flexion with the actuator saturation handler in AP-prioritization) is repeated but with the addition of the inverse plant controller. This allows for a fair comparison and highlights the effect of the inverse plant controller. Figure 5.7 and 5.8 show the result of addition of the inverse plant controller.

Directly it is visible that the actuators are active from $t = 0$, as seen in Figure 5.8. This can also be seen in the motion, as there is already recovery of the ALC deficient knee with soft robotic brace motion from the ACL deficient knee motion towards the ACL intact knee motion in the first two seconds. When time progresses, the ILC will learn from the difference which is left and compensate accordingly. This initial control signal also benefits the recovery at the end of the simulation. The recovery in the AP displacement is 76.2% and in the IE rotation 37.6%. An improvement of 7.9% and 13.0% respectively over not using the inverse plant controller. The inverse plant controller has also bee evaluated in IE-prioritization (depicted in appendix D). Here a IE recovery of 64.3% was measured at a lower AP recovery of 40.0%. An improvement of 10.1% in the IE rotation over the experiment performed in Section 5.3, but 12.3% worse

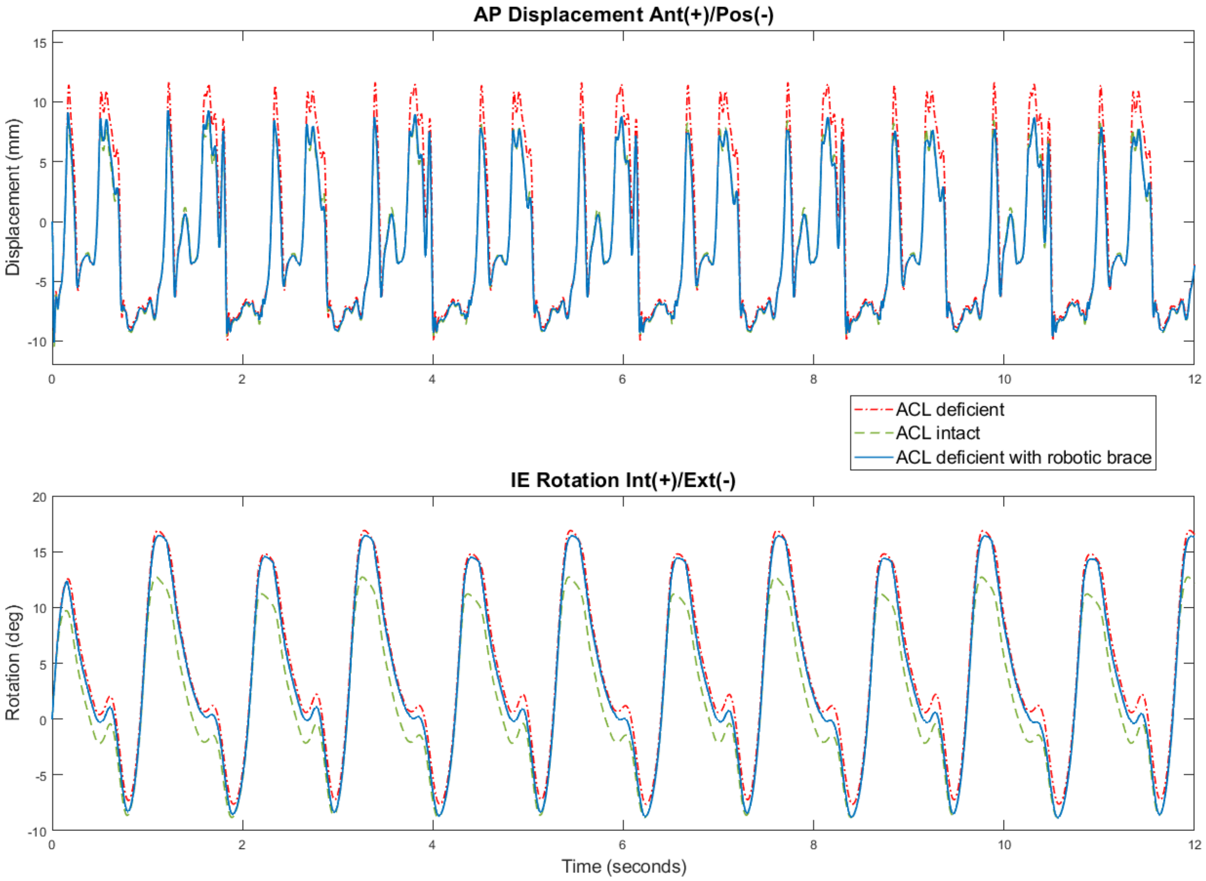

Figure 5.7: Inverse plant controller addition results at 30 degrees of discrete knee flexion. AP-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

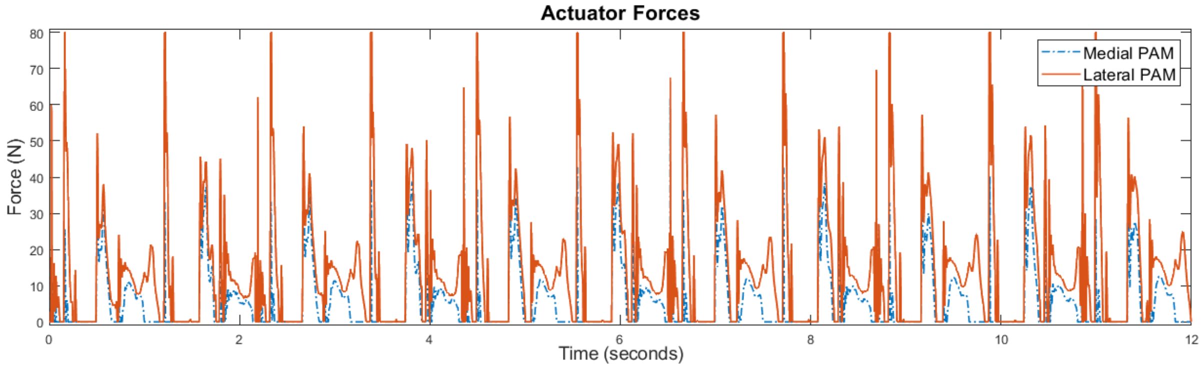

Figure 5.8: Actuator forces for the motion result in Figure 5.7

recovery in the AP displacement. The worsened AP recovery in IE-prioritization is expected, as explained at the end of Section 5.3. While the IE control signal is optimized while in IE-prioritization mode, AP recovery might suffer to make room for greater IE recovery, and vice versa.

As the controller is in AP-prioritization it is expected to see the most convergence in the AP displacement, but looking at the IE rotation it can be seen that around 0° rotation there is also convergence towards ACL intact motion of the ACL deficient knee with soft robotic brace. At the larger internal angles, around 10° to 15°, there is almost no convergence. This can be related back to hardware limitations of the actuator forces, while in the first case there is room in increase the difference between the actuator forces while keeping the average the same, this is not possible at the larger internal angles. At these larger internal angles more external torque is applied as a result of the gait motion seen in Figure 4.3.

In this case the inverse plant controller shows promising results. However, an error in inverse mapping might give off a too large of a force. For a robotic solution used in hybrid with a human joint this can be dangerous. Not using an inverse plant controller means that the ILC has to learn more, as shown in Figure 5.1 compared to Figure 5.7. It will take the ILC more iterations to convergence to ACL intact motion, but it will be safer.

Another drawback is that this inverse plant still uses state feedback to determine the current state of the plant. This might cause instabilities in the real world. A solution to this in the real world might be determine the state by means of a model. This way the feedback can be omitted at the cost of computing power.

5.5 PAM Robustness

To evaluate the effect of the slowness of the PAMs the input force to the ideal prismatic actuators in the model has been rate-limited. This introduces a maximum rising and falling slew rate. As the exact characteristics of the PAMs are not known, different rising and falling slew rate limits (equal for rising and falling, in maximum actuator force (80N) per seconds N_{max}/s. 2 N_{max}/s means the actuator can deliver its maximum actuator force of 80N and go back to 0N in 1 second) are evaluated and their recovery is measured.

Figure 5.9 shows the recovery for AP and IE motion for the last two seconds of the simulation, from the 10 seconds mark to the 12 seconds mark, in discrete 30° of knee flexion with the inverse plant controller active and the actuator saturation handler in AP-prioritization.

It can be seen that with increased maximum actuation speed, slew rate, the AP recovery improves as expected. For IE it can be seen that from a slew rate of 1 to 4 N_{max}/s the recovery decreases, however, when increasing to 8 N_{max}/s and higher the recovery increases again. This might indicate that there is are multiple local optimum between PAM damping and the knee model. To get a clear result of the effect of the rate limit on the PAMs on the IE rotation the same experiment has to be performed in IE-prioritization as well.

5.6 Controller Performance During Human Gait

After the individual controller stages have been determined, the controller is configured to be used during gait. This is done by fitting a linear interpolated surface to the four discrete inverse plant curves, as explained in Section 4.4.1. The ILC is configured with the same learning gains of $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$. For the actuator saturation handler, it has been set to AP-prioritization in the first simulation and IE-prioritization in the second simulation.

Figures 5.10 and 5.11 show the experimentation of the above controller, in either prioritization mode, in combination with the soft robotic knee brace on an ACL deficient knee model with continuous knee flexion angle. This experiment gives a good impression how the control system will actuate the soft robotic knee brace during gait.

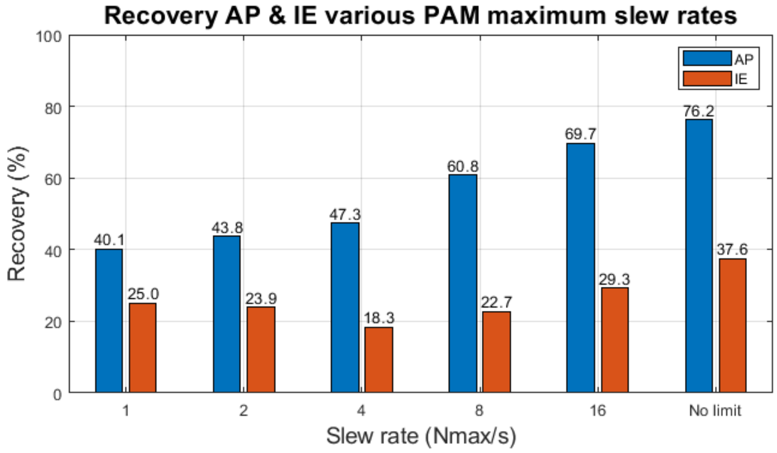

Figure 5.9: Recovery percentages for different actuator slew rates. AP-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

Timeframe	[0 2] seconds	[10 12] seconds
AP $_{\text{AP-prioritization}}$	60.5%	75.5%
AP $_{\text{IE-prioritization}}$	35.9%	49.5%
IE $_{\text{AP-prioritization}}$	16.1%	19.5%
IE $_{\text{IE-prioritization}}$	34.4%	36.8%

Table 5.1: Recovery percentages of AP and IE motion during gait of the ACL deficient knee model with soft robotic knee brace in AP and IE-prioritization during the first two and last two seconds of the simulation

The results in Table 5.1 show an already present recovery at the first two seconds of the simulation as a result of the inverse plant controller. In the last two seconds it can be seen that there is an improvement in recovery over the first two seconds as a result of learning performed by the AP and IE ILC. It can furthermore be seen that the AP displacement in AP-prioritization receives a larger recovery compared to the AP displacement in IE-prioritization. The same can be seen for IE rotation in IE-prioritization and AP-prioritization.

When visually inspecting the recovery in Figure 5.10 in AP-prioritization the recovery numbers can be visually confirmed. In the AP translation a clear and large recovery is visible. In the IE rotation only a recovery is seen during the double support at the end of the stance phase (45% to 60% of the gait cycle), confirming the low overall recovery numbers for IE rotation in AP-prioritization.

Inspecting the recovery in Figure 5.11 it can be observed that the AP recovery is less compared to the AP-prioritization. The lower recovery number in the AP displacement is caused by the lesser recovery when transitioning from the end of the swing phase back to the stance phase of the gait cycle. Seen, for example, at $t = 10.5s$ to $t = 11.0s$. At $t = 11.0s$ at double support of the gait cycle (0% to 10%) some over compensation in the AP displacement can be seen, here the soft robotic brace over compensated the AP motion to result in more recovery in the IE motion. In the same figure it can also be seen that the IE rotation when the knee flexion angle is between 10° and 20° the recovery is not optimal and the ACL deficient knee with soft robotic brace motion still deviates significantly from the ACL intact motion. This is during the beginning of the stance phase (0% to approximately 30% of the gait cycle, as of Figure 3.5) and the end of the swing phase (95% to 100% of the gait cycle).

This significant deviation from ACL intact motion can be caused by a multiple of reasons. The first reason could be traced back to the ACL deficient knee model in which between 0° to 20° of knee flexion the IE surface fit behaves linearly, as shown in Figure 3.11. This can be improved by having introducing laxity data for knee flexion angles between 0° and 30° to validate this result. The second reason can be derived from the angle at which the PAMs will deliver forces to the tibial support plate at these knee flexion angles. The PAMs at the tibial support plate are at an angle at which the majority of the forces are not in the direction of the IE rotation of the tibia.

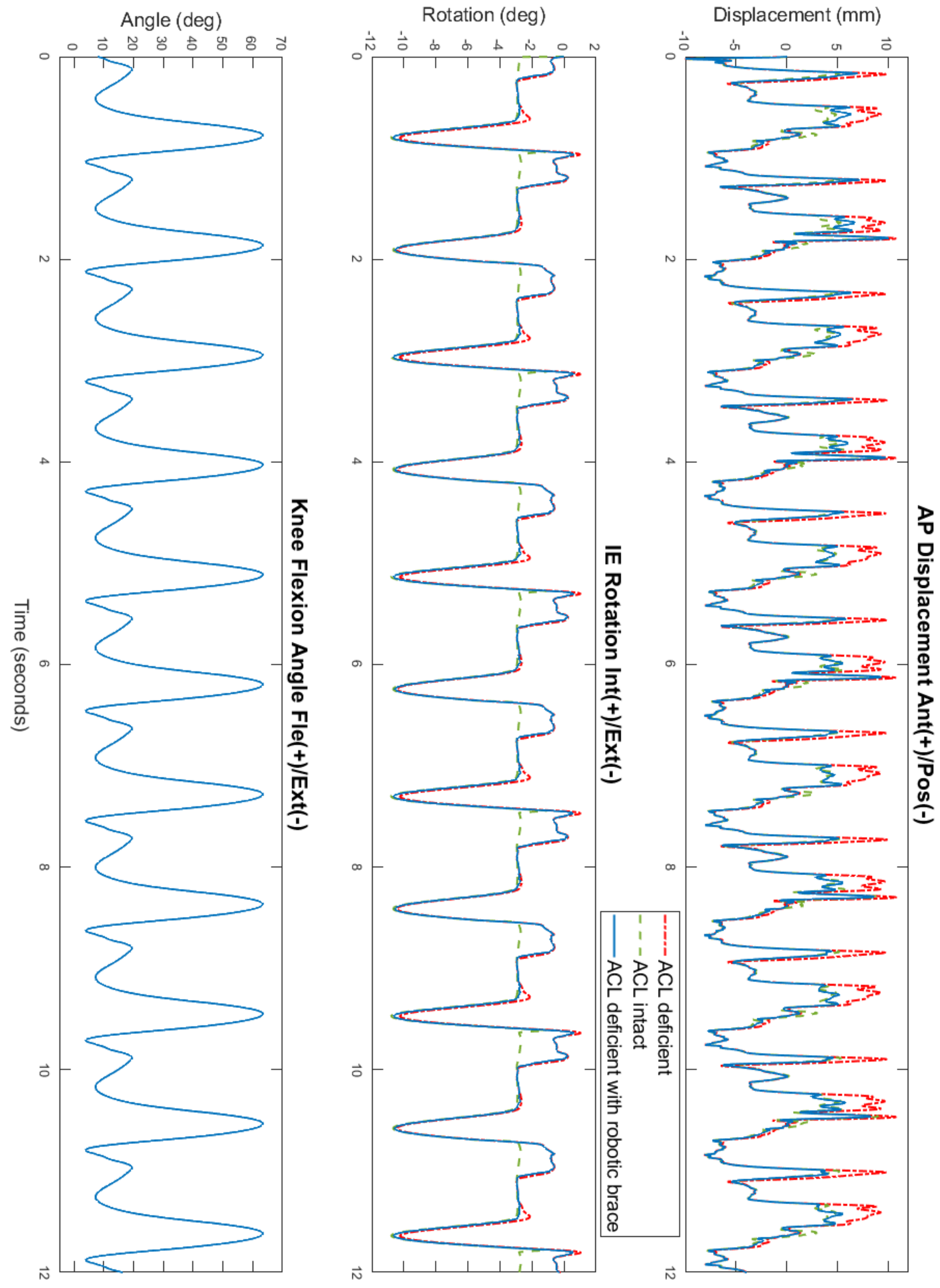

Figure 5.10: Knee motion during gait with the use of the soft robotic knee brace in AP-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

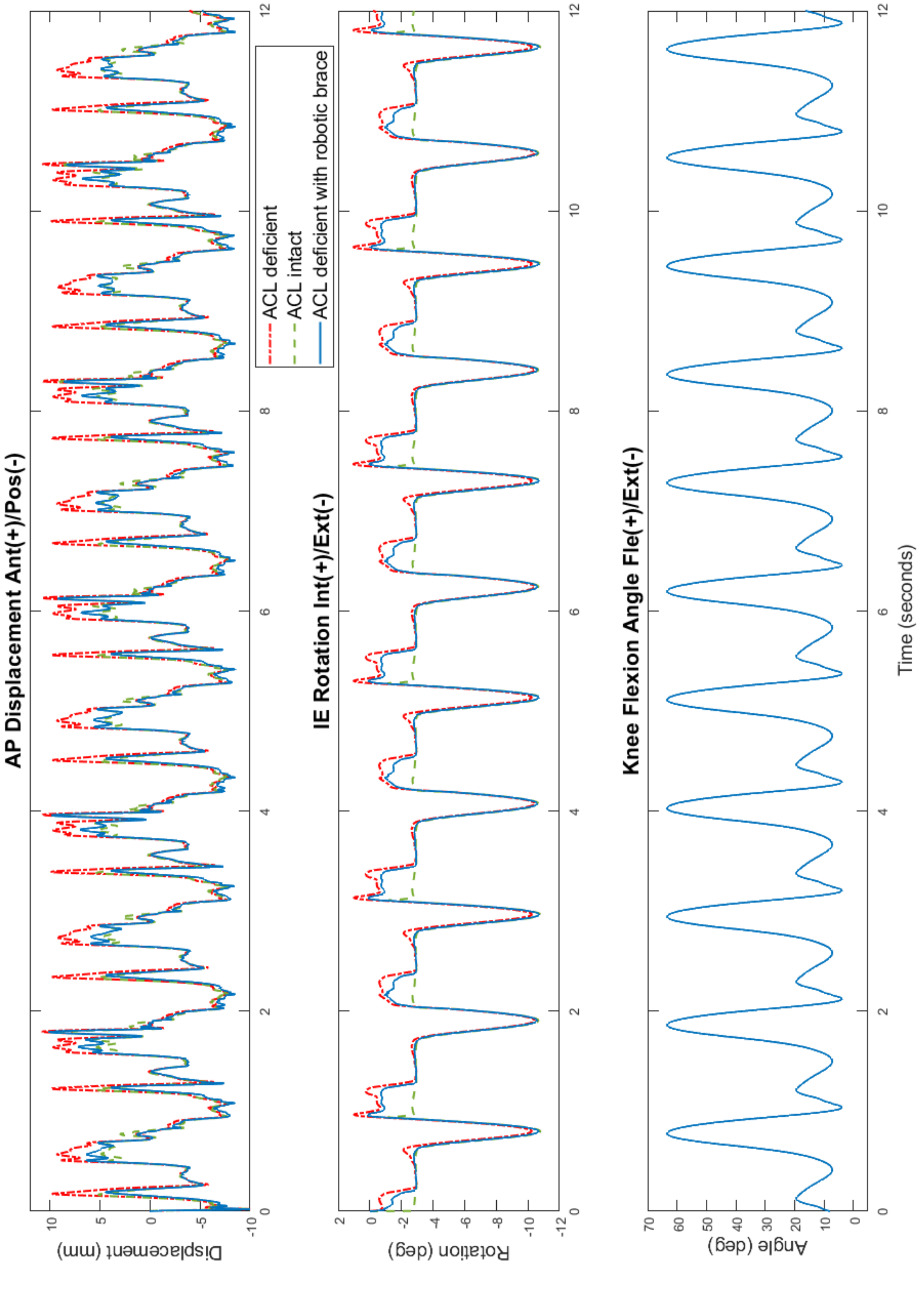

Figure 5.11: Knee motion during gait with the use of the soft robotic knee brace in IE-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

6 Conclusions

The goal of this research is to return healthy motion to an ACL deficient knee tailored to the patient during gait. This has been realized by developing a control system for the already existing soft robotic knee brace and validating on a knee model. The results highlight that it is possible to (partially) bring back ACL intact motion, and therefore reduce the increase in laxity caused by ACL lesion.

A second goal is to achieve the above while not obstructing any healthy motion in any way. Depending on the prioritization of the actuator saturation handler the system can prioritize AP or IE motion, but this can be at the cost of the other. This might obstruct healthy motion, such as shown in Figure 5.11 where the AP displacement is over compensated to realize the IE recovery in some small regions. It is expected that in the current state of the control design the system can obstruct healthy motion in the AP-displacement when in IE-prioritization, and vice versa. This is however the latter is not validated in simulations.

From the experiments of the individual controller stages it can be concluded that the iterative learning controller is capable of learning from the difference between healthy, ACL intact, motion and the ACL deficient knee model motion. Resulting in an AP recovery of 68.3% (and 24.6% IE recovery) in AP-prioritization and an IE recovery of 54.2% (and 52.3% AP recovery) in IE-prioritization in discrete knee flexion angle experiments.

The addition of the inverse plant controller is able to improve the final recovery by supplying an initial control signal as shown in the discrete knee flexion angle experiments. In AP-prioritization an improved AP recovery of 76.2% and IE recovery of 37.6% was obtained. An improvement of 7.9% and 13.0% respectively over constructing the control signal with the ILC alone. In IE-prioritization an IE recovery of 64.3% is obtained, an improvement of 7.9%. However, a lower AP recovery of 40.0% was obtained in IE-prioritization. A decrease of 12.3% in recovery caused by the IE-prioritization mode with a more optimal IE control signal.

The actuator saturation handler is successfully able to prioritize either AP or IE motion, or a weighted sum between them, in case if one (or two) actuators saturate. The recovery is directly related to the AP-IE prioritization weight. This allows the clinician to tailor the behaviour of the soft robotic brace to the patient.

The PAM characteristics were not known during this research, however an indication experiment has been performed to show the robustness of the control system on different PAM input-output rates, implemented in slew rates. It was shown that the recovery in both AP and IE motion was negatively affected with lower slew rates.

During gait simulations the control system performed as expected on the soft robotic brace and was able to return a recovery of 75.5% in the AP displacement and 19.5% in IE rotation in AP-prioritization. In IE-prioritization a higher recovery in IE rotation was obtained of 36.8% at a cost of a lower recovery of 49.5% in the AP displacement. The least recovery was found to be in IE rotation during the end of the swing phase till half way through the stance phase when the knee flexion angle was between 10° and 20°.

The transition of non-weightbearing to weightbearing was not and cannot be examined using the current knee model in this research. The same can be said for the effect on proprioception. For the latter in vivo experiments are necessary. In the end the research focuses solely on reducing the laxity caused by ACL lesion.

7 Shortcomings & Future Research

This study focused on designing and validating a control system for a soft robotic knee brace with the focus on treating ACL lesion. Future research can be subdivided into three categories: improving the ACL intact and deficient knee model, validating the control on real world soft robotic knee brace, improving the control design with a more advanced system.

Due to the COVID-19 pandemic the necessary labs in the Netherlands were closed. As a result there was no possibility to perform real world laxity experiments to create a more accurate knee model and validate the knee model. Neither was it possible to validate the workings of the control system on the real world brace. Performing laxity experiments at more flexion angles and with more loads would allow the creation of a more accurate knee model. Dynamic system identification would allow the creation of an accurate dynamical model of the knee. The latter would also open the possibilities of dynamic control.

To improve the soft robotic brace model two main improvements can be done: the interaction of brace and skin must be modelled and the (dynamic) characteristics of the PAMs must be identified. Especially the latter would improve the results of this research.

Besides the above mentioned shortcoming of the model due to limited laxity data, the current model can be extended with valgus-varus and superior-inferior motion characteristics. The first would allow the control design to be extended to possible cover the increase in valgus-varus laxity due to ACL lesion. The latter would allow experimentation with the transition of non-weightbearing to weightbearing. The coupled behaviour of AP, IE, VV, and SI could also be added as currently AP and IE are decoupled.

The current controller design also comes with some shortcomings and possible improvements. Currently the timing of the ILC iteration time is set by hand and is derived from already knowing the gait cycle timing. A detection can be implemented that dynamically adjusts this timing. The ILC furthermore introduces integral behaviour and suffers from windup if the error persists.

As mentioned in the results, the inverse plant is highly dependent on the configuration of the plant, meaning the soft robotic brace and knee. An error could lead to unforeseen actuation which could in turn lead to decreased patient well being. The compliant behaviour of the PAMs must prevent injury in such a case. The inverse plant also is state-dependent, meaning it requires state-feedback. This could also cause instability.

Lastly, the values of γ_{AP}, γ_{IE}, and the AP-IE prioritization value r are varied in discrete flexion angles simulations. For a more real world view these values should also be varied in simulations with continuous knee flexion angles to get a better understanding how they would influence the soft robotic knee brace during gait.

A Appendix: Experimentation Setup

A.1 Introduction

This appendix will explain on how to prepare the experiments in Simulink and how to repeat the experiments. For each file description please refer to the README.txt included with the files.

A.2 Requirements

- MATLAB R2020a (or higher)
- Simulink version 10.1 (R2020a) (or higher)
- Simscape version 4.8 (or higher)
- Simscape Driveline version 3.1 (or higher)
- Simscape Electrical version 7.3 (or higher)
- Simscape Fluids version 3.0 (or higher)
- Simscape Multibody version 7.1 (or higher)
- Curve Fitting Toolbox version 3.5.11 (or higher)
- SLM - Shape Language Modeling version 1.14 (or higher) by John D'Errico available at the MatWorks File Exchange

A.3 File overview

```
\Data Generators\GaitCycleFlexionExtension.m
\Data Generators\loadingDataGenerator.m
\Data Generators\loadingDataGeneratorC926B_R.m
\Profilers\APForceProfilerACL.m
\Profilers\APForceProfilerACLD.m
\Profilers\FFAPForceProfilerACLD.m
\Profilers\FFidentification.m
\Profilers\FFIETorqueProfilerACLD.m
\Profilers\IEForceProfilerACL.m
\Profilers\IEForceProfilerACLD.m
APForceFitData.mat
APForceFitDataACL.mat
APForceFunc.m
APForceFuncACL.m
APForceInterpolalatedSurfaceFitACLData.mat
APForceInterpolalatedSurfaceFitData.mat
APIEModel_Characterized_ILC_Comparison.slx
extendedKneeAngleData.mat
FFAPForceFitData.mat
FFAPForceInterpolatedSurfaceFitData.mat
FFAPInverseFunc.m
FFIEInverseFunc.m
FFIETorqueFitData.mat
FFIETorqueInterpolatedSurfaceFitData.mat
```

```
IETorqueFitData.mat
IETorqueFitDataACL.mat
IETorqueFunc.m
IETorqueFuncACL.m
IETorqueInterpolalatedSurfaceFitACLData.mat
IETorqueInterpolalatedSurfaceFitData.mat
loadingDataAP.mat
loadingDataAPC926B_L.mat
loadingDataExtender.m
loadingDataIE.mat
loadingDataIEC926B_L.mat
main.APIEModel.m
README.txt
```

A.4 Installation

The user has to download the *APIEModelControl.zip*, which can be obtained by contacting the author, and extract the .zip file.

A.5 Running the simulation

The user must open *APIEModel_Characterized_ILC_Comparison.slx* and run the *main_APIEModel.m* file. In the MATLAB command window the user can specify experimentation with discrete knee flexion angles or with a continuous knee flexion angle (gait knee flexion angles). For the first the user is asked to specify the flexion angle with the choice of: 0°, 30°, 60°, or 90°. The Simulink model can now be run by clicking *run* in the Simulink window.

A.6 Customization the loads

The user can specify the loads during the simulation by setting the desired AP and IE loads in the external forces and disturbances area in the Simulink plant (*APIEModel_Characterized_ILC_Comparison/ACL deficient Knee Model with Soft Robotic Brace* subsystem, it will also apply the the ACL intact and ACL deficient model for comparison), as explained in Section 3.5. By default gait load force profile is applied.

B Appendix: Laxity Data Graphs

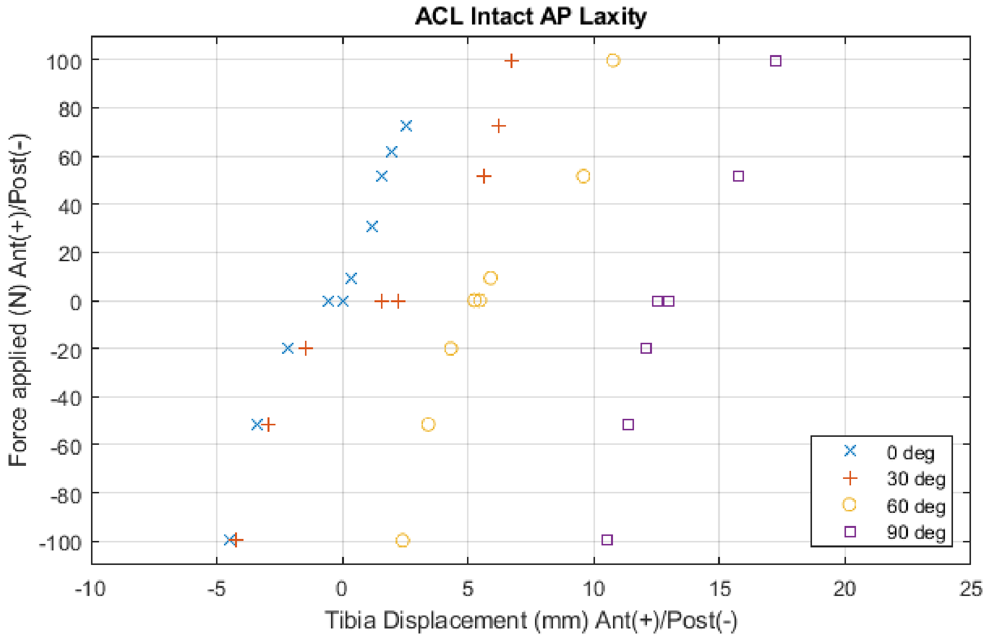

Figure B.1: ACL intact AP laxity data

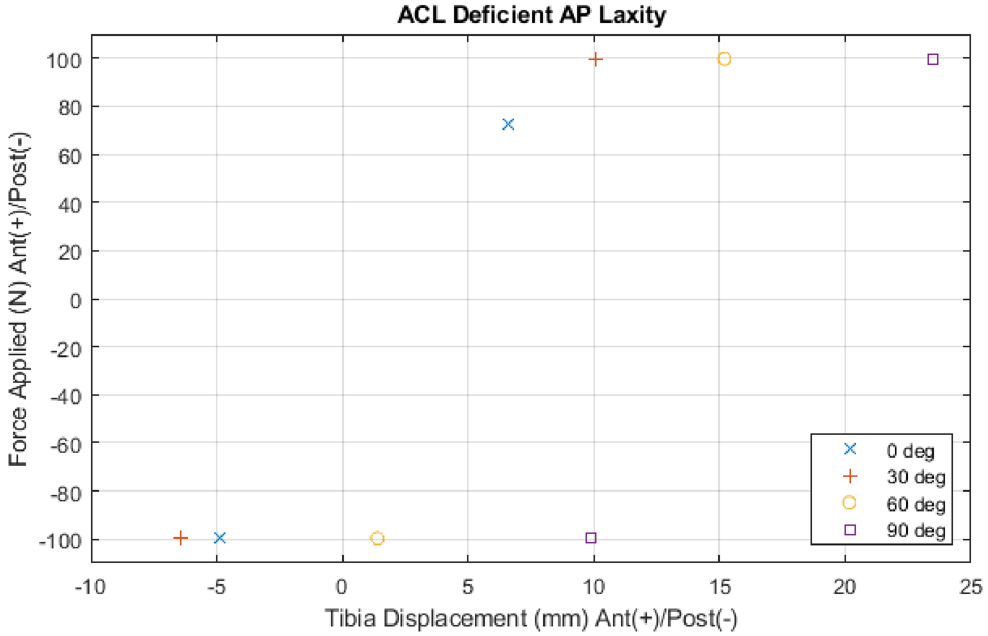

Figure B.2: ACL deficient AP laxity data

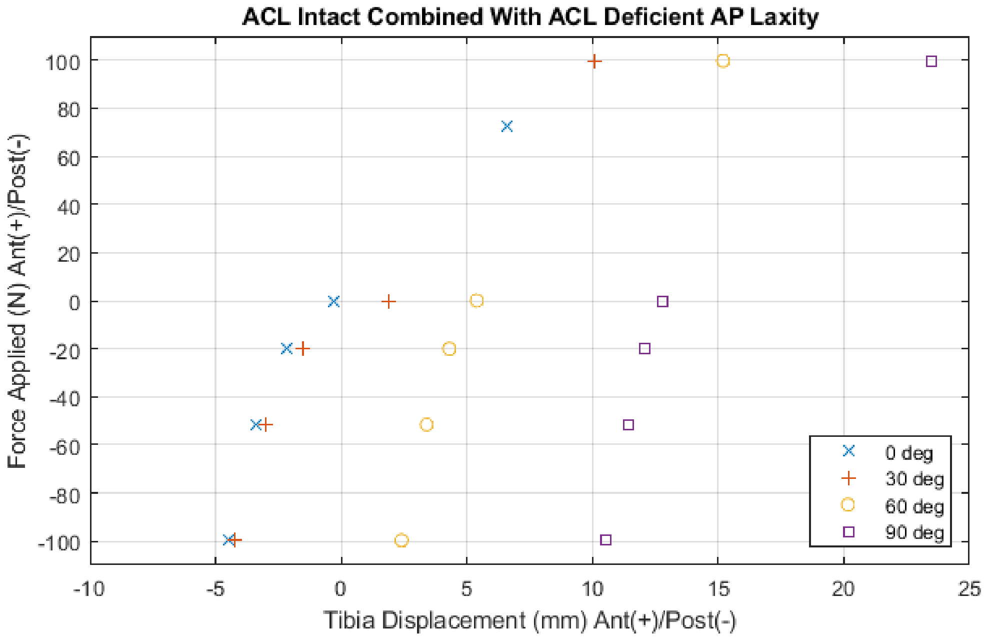

Figure B.3: ACL intact AP laxity data combined with ACL deficient AP laxity data

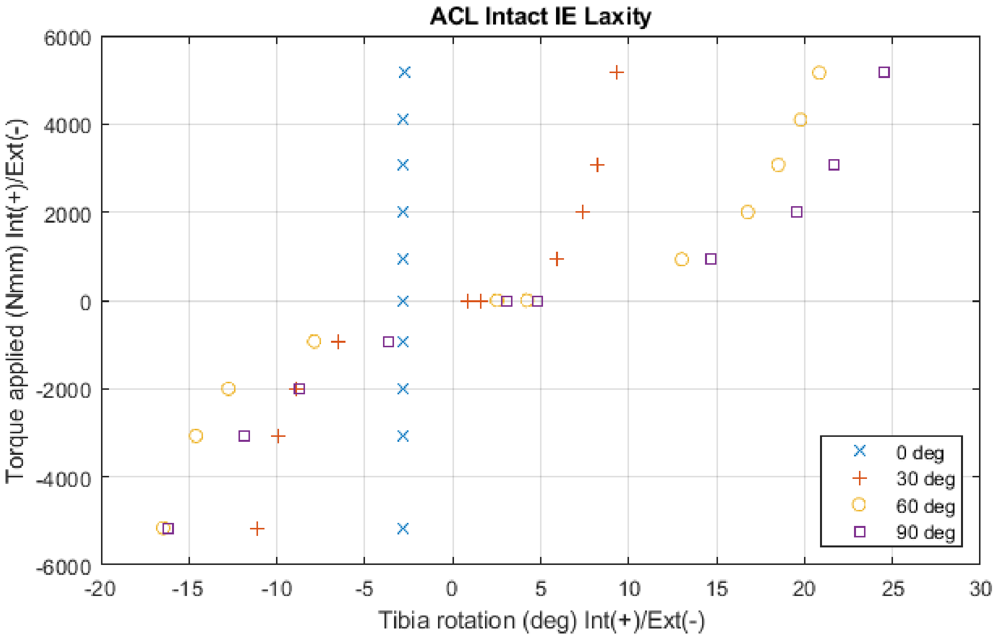

Figure B.4: ACL intact IE laxity data

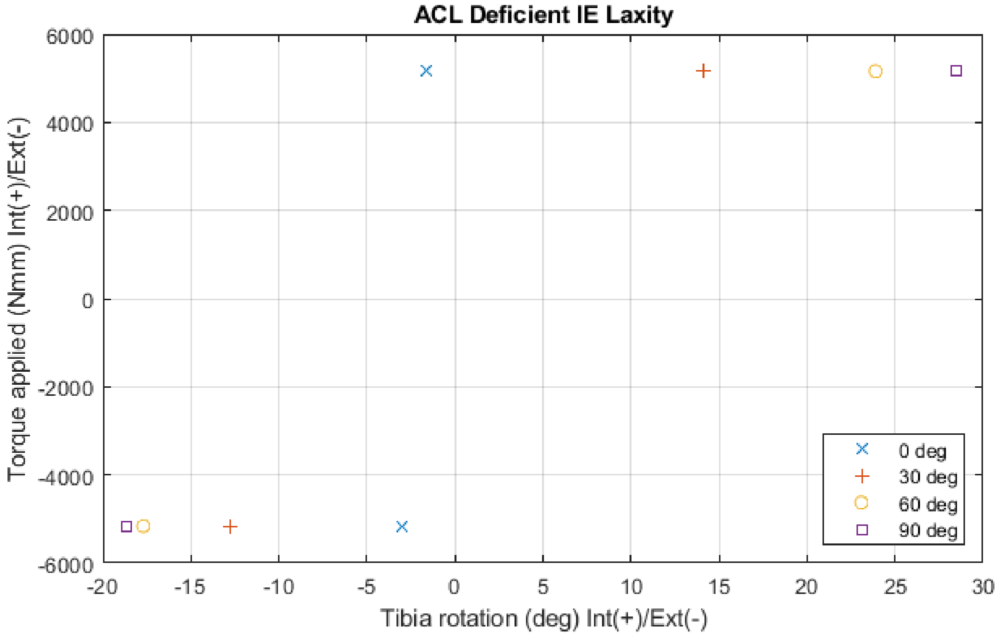

Figure B.5: ACL deficient IE laxity data

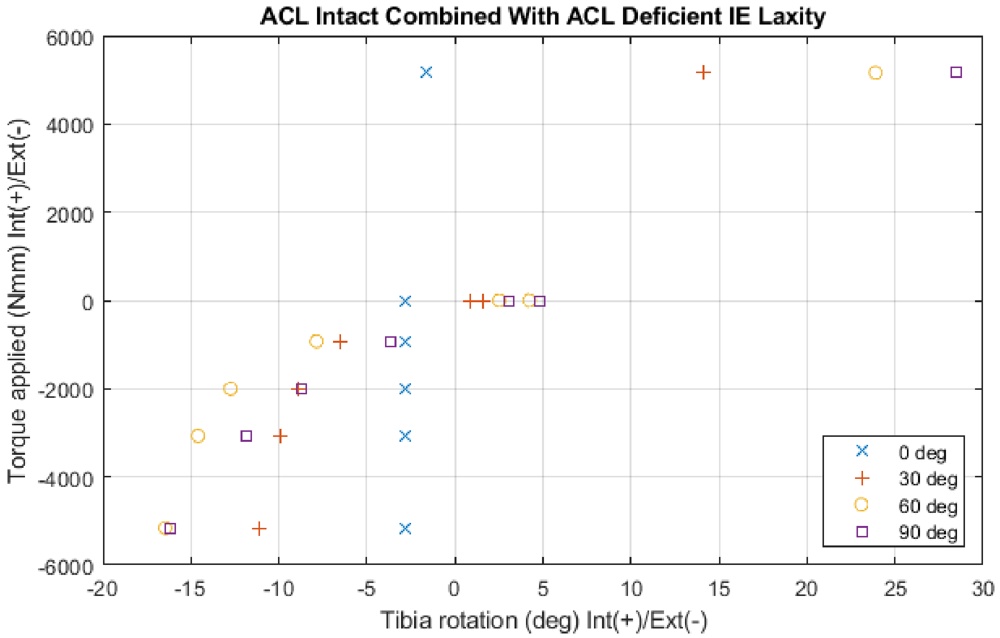

Figure B.6: ACL intact IE laxity data combined with ACL deficient IE laxity data

C Appendix: ILC Performance for Different Learning Gains

Please refer to the next two pages for the graphs

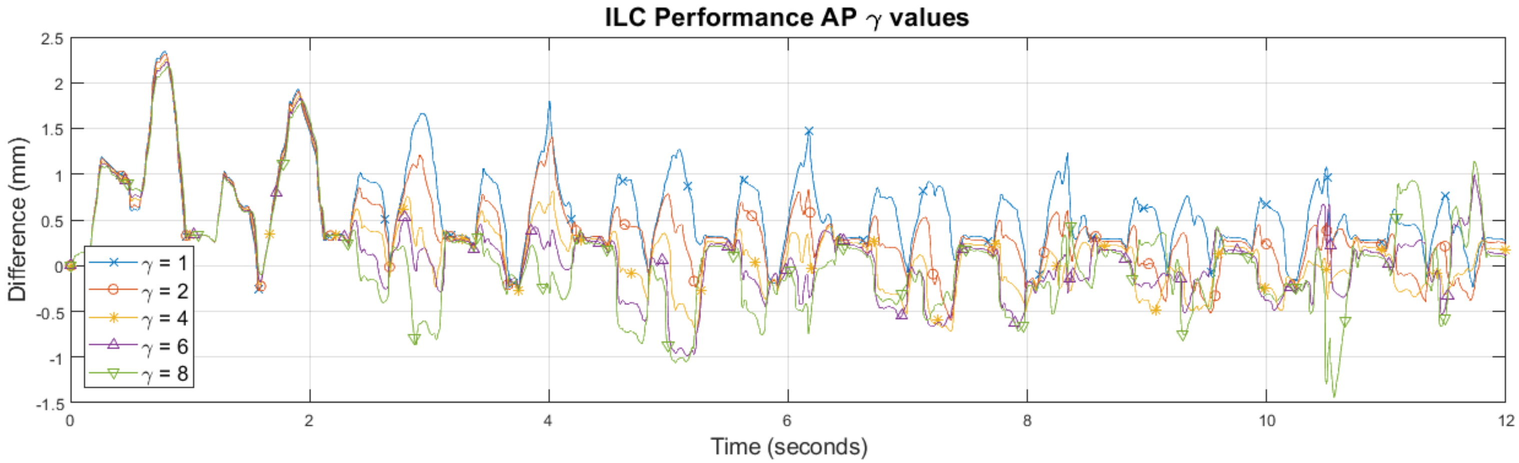

Figure C.1: AP displacement difference between ACL intact motion and ACL deficient with soft robotic brace motion. Data shown in a trailing moving average of 0.25 seconds.

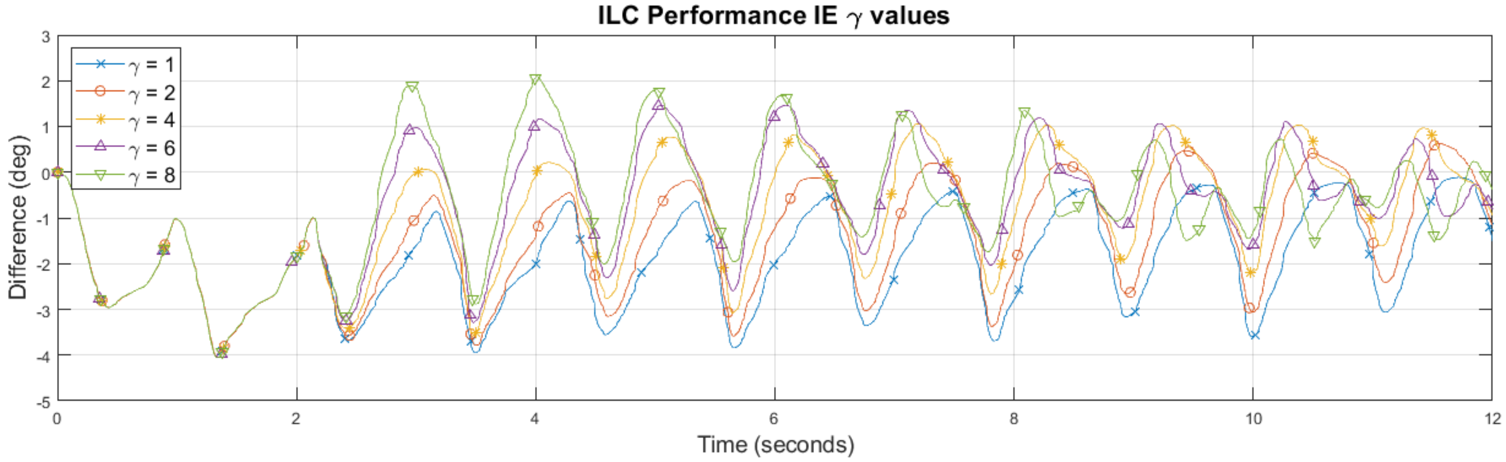

Figure C.2: IE rotation difference between ACL intact motion and ACL deficient with soft robotic brace motion. Data shown in a trailing moving average of 0.25 seconds.

D Appendix: Inverse Plant Controller Performance in IE-Prioritization

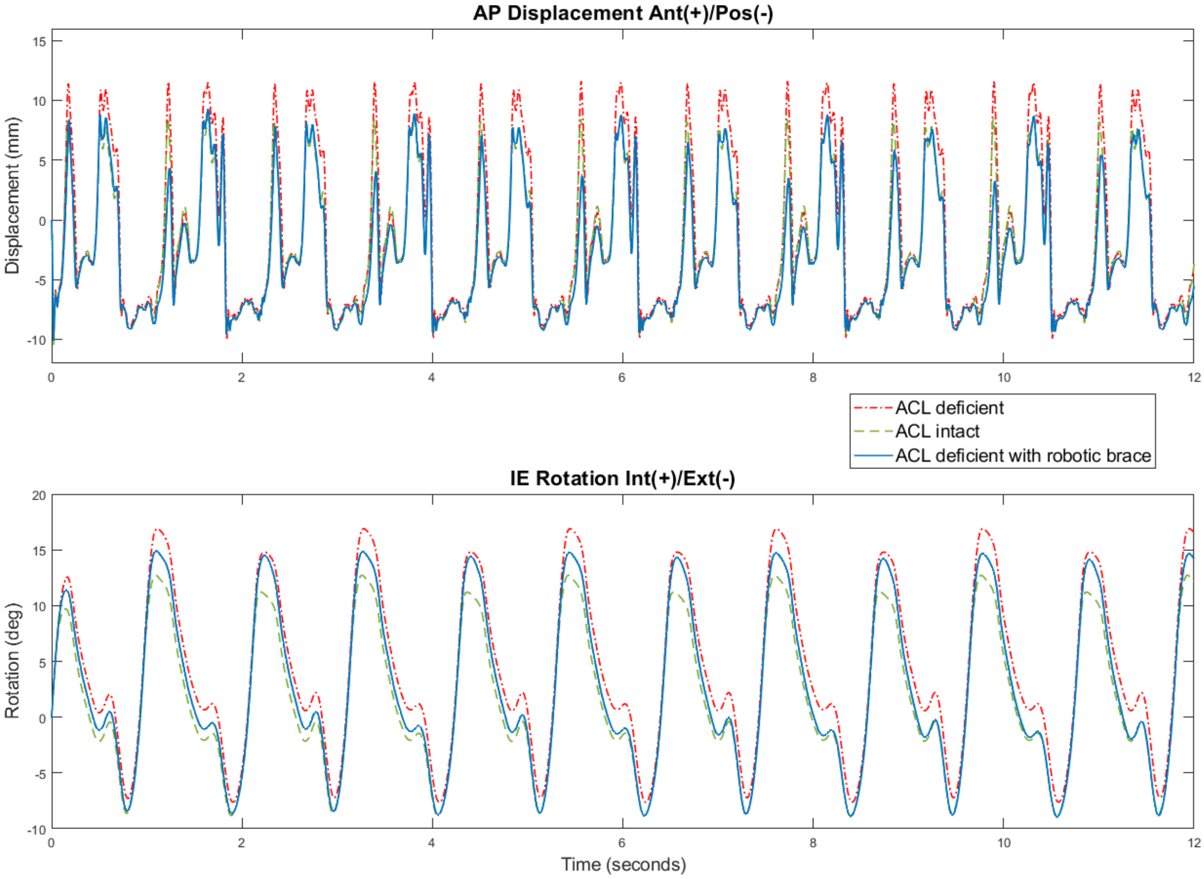

Figure D.1: Inverse plant controller addition results at 30 degrees of discrete knee flexion. IE-prioritization. $\gamma_{AP} = 4.0$ and $\gamma_{IE} = 1.0$

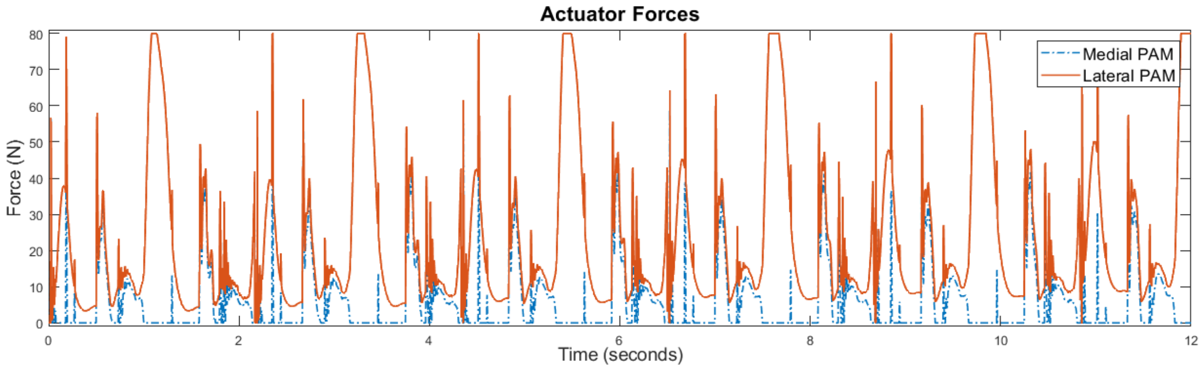

Figure D.2: Actuator forces for the motion result in figure D.1

E Appendix: Actuator Saturation Handler MATLAB code

```
function [r, state, Mout, Lout, avgForceOut, diffForceOut] =
    StateMachineForceRegulator(Min, Lin)

%% Inputs
% Min: medial input force control signal
% Lin: laterial input force control signal

%% Outputs
% r: prioritization value
% state: current state (debugging)
% Mout: medial force control signal output
% Lout: llateral force control signal output
% avgForceOut: average Force Out (debugging)
% diffForceOut: difference force out (debugging)

%% Initial parameters
r = 1;               % ratio of AP or IE priorisation. 1.0 =  full AP,
    0.0 full IE.
maxForcePAM = 80;    % maximum force deliverable by the PAM
minForcePAM = 0;     % minimum force deliverable by the PAM
maxForceDiff = abs(maxForcePAM)-abs(minForcePAM);

%% Clarification
% AP is directly linked to the average of the non-corrected (raw)
    signals.
% IE is directly linked to the difference of the non-corrected signals.

%% State machine

avgForce = (Min + Lin)/2;    % in AP priorisation this average is kept
    after compensation
diffForce = abs(Min - Lin); % in IE priorisation this difference is kept
     after compensation

% AP is recoverable if avgForce is minForcePAM <= avgForce <=
    maxForcePAM
% IE is recoverable if diffForce is minForcePAM <= diffForce <=
    maxForcePAM

% state 0: all forces are in operating range
if Min >= minForcePAM && Min <= maxForcePAM && Lin >= minForcePAM && Lin
     <= maxForcePAM
    Mout = Min;
    Lout = Lin;
    state = int8(0);

% state 1: Medial force saturates (max), lateral force does not
elseif Min > maxForcePAM && Lin <= maxForcePAM && Lin >= minForcePAM
    Mout = maxForcePAM;
    Lout = (1.0-r)*(Mout - diffForce) + r*(avgForce - abs(Mout-avgForce)
        );
    Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if Lout
        becomes too large/small
    state = int8(1);

% state 2: Medial force saturates (min), lateral force does not
elseif Min < minForcePAM && Lin >= minForcePAM && Lin <= maxForcePAM
    Mout = minForcePAM;
```

```
        Lout = (1.0-r)*(Mout + diffForce) + r*(avgForce + abs(Mout-avgForce)
            );
        Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if Lout
            becomes too large/small
        state = int8(2);

    % state 3/4: Both medial and lateral forces saturate (min and max)
    elseif Min > maxForcePAM && Lin < minForcePAM
        if abs(Min - maxForcePAM) >= abs(Lin - minForcePAM)
            Mout = maxForcePAM;
            Lout = (1.0-r)*(Mout - diffForce) + r*(avgForce - abs(Mout-
                avgForce));
            Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if
                Lout becomes too large/small
            state = int8(3);

        else
            Lout = minForcePAM;
            Mout = (1.0-r)*(Lout + diffForce) + r*(avgForce + abs(Lout-
                avgForce));
            Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if
                Lout becomes too large/small
            state = int8(4);

        end

    % state 5: Lateral force saturates (max), medial force does not
    elseif Lin > maxForcePAM && Min <= maxForcePAM && Min >= minForcePAM
        Lout = maxForcePAM;
        Mout = (1.0-r)*(Lout - diffForce) + r*(avgForce - abs(Lout-avgForce)
            );
        Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if Mout
            becomes too large/small
        state = int8(5);

    % state 6: Lateral force saturates (min), medial force does not
    elseif Lin < minForcePAM && Min >= minForcePAM && Min <= maxForcePAM
        Lout = minForcePAM;
        Mout = (1.0-r)*(Lout + diffForce) + r*(avgForce + abs(Lout-avgForce)
            );
        Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if Lout
            becomes too large/small
        state = int8(6);

    % state 7/8: both medial and lateral forces saturate (min and max)
    elseif Lin > maxForcePAM && Min < minForcePAM
        if abs(Lin - maxForcePAM) >= abs(Min - minForcePAM)
            Lout = maxForcePAM;
            Mout = (1.0-r)*(Lout - diffForce) + r*(avgForce - abs(Lout-
                avgForce));
            Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if
                Mout becomes too large/small
            state = int8(7);

        else
            Mout = minForcePAM;
            Lout = (1.0-r)*(Mout + diffForce) + r*(avgForce + abs(Mout-
                avgForce));
            Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if
                Lout becomes too large/small
            state = int8(8);

        end
```

```
% state 9/10/11: both medial and lateral forces saturate (both max)
elseif Min > maxForcePAM && Lin > maxForcePAM
    if Min > Lin
        Mout = maxForcePAM;
        Lout = (1.0-r)*(Mout - diffForce) + r*(avgForce - abs(Mout-
            avgForce));
        Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if
            Lout becomes too large/small
        state = int8(9);

    elseif Lin > Min
        Lout = maxForcePAM;
        Mout = (1.0-r)*(Lout - diffForce) + r*(avgForce - abs(Lout-
            avgForce));
        Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if
            Mout becomes too large/small
        state = int8(10);

    else
        Mout = maxForcePAM;
        Lout = maxForcePAM;
        state = int8(11);

    end

% state 12/13/14: both medial and lateral forces saturate (both min)
elseif Min < minForcePAM && Lin < minForcePAM
    if Min < Lin
        Mout = minForcePAM;
        Lout = (1.0-r)*(Mout + diffForce) + r*(avgForce + abs(Mout-
            avgForce));
        Lout = min(maxForcePAM, max(minForcePAM, Lout)); % Saturate if
            Lout becomes too large/small
        state = int8(12);

    elseif Lin < Min
        Lout = minForcePAM;
        Mout = (1.0-r)*(Lout + diffForce) + r*(avgForce + abs(Lout-
            avgForce));
        Mout = min(maxForcePAM, max(minForcePAM, Mout)); % Saturate if
            Mout becomes too large/small
        state = int8(13);

    else
        Mout = minForcePAM;
        Lout = minForcePAM;
        state = int8(14);

    end

% error state
else
    Mout = 0;
    Lout = 0;
    state = int8(-1);

end

% measuring variables
avgForceOut = (Mout + Lout)/2;      % in AP priorisation this average is
     kept after compensation
```

```
diffForceOut = Mout - Lout;         % in IE priorisation this difference
     is kept after compensation
```

F Appendix: MATLAB Code ACL Intact Data Fit

F.1 AP Force

```
%% Data & Fitting
% 0 deg
%           [N]     [mm]            [deg]
APdata0 =  [72.716      2.534110668     0
            61.936      1.953006261     0
            51.597      1.543661857     0
            30.723      1.204733603     0
            9.31        0.333201039     0
            0       0               0
            0       -0.528748403        0
            -20.041     -2.179811623    0
            -51.597     -3.41966231     0
            -99.764     -4.504701854    0];

xAP0 = APdata0(:,2);
yAP0 = APdata0(:,3);
zAP0 = APdata0(:,1);

APForceFit0 = slmengine(xAP0,zAP0,'knots',[-20 20],'increasing','on','
    result','pp');

% 30 deg
APdata30 = [99.764      6.70494176      30
            72.716      6.187120207     30
            51.597      5.638011076     30
            0       2.194401286     30
            0       1.580729592     30
            -20.041     -1.48079202     30
            -51.597     -2.973666327    30
            -99.764     -4.219763874    30];

xAP30 = APdata30(:,2);
yAP30 = APdata30(:,3);
zAP30 = APdata30(:,1);

APForceFit30 = slmengine(xAP30,zAP30,'knots',[-25 30],'increasing','on',
    'result','pp');

% 60 deg
APdata60 = [99.764      10.76376669     60
            51.597      9.585668796     60
            9.31        5.893966425     60
            0       5.471167156 60
            0       5.257743444 60
            -20.041     4.310045159     60
            -51.597     3.421742761     60
            -99.764     2.389301091     60];

xAP60 = APdata60(:,2);
yAP60 = APdata60(:,3);
zAP60 = APdata60(:,1);

APForceFit60 = slmengine(xAP60,zAP60,'knots',[-25 30],'increasing','on',
    'result','pp');

% 90 deg
APdata90 = [99.764      17.21149118     90
```

```
            51.597      15.72735776     90
            0       13.00863747 90
            0       12.54095969 90
            -20.041     12.05655533     90
            -51.597     11.37207831     90
            -99.764     10.53985749     90];

xAP90 = APdata90(:,2);
yAP90 = APdata90(:,3);
zAP90 = APdata90(:,1);

APForceFit90 = slmengine(xAP90,zAP90,'knots',[-10 40],'increasing','on',
    'result','pp');

%% Interpolate points from curve fits to be used in surface fit
APForceFitInterpolated0     = ppval(APForceFit0, -40:1:50);
APForceFitInterpolated30    = ppval(APForceFit30, -40:1:50);
APForceFitInterpolated60    = ppval(APForceFit60, -40:1:50);
APForceFitInterpolated90    = ppval(APForceFit90, -40:1:50);

xData = cat(1,-40:1:50,-40:1:50,-40:1:50,-40:1:50); % displacement [mm]
yData = cat(1,0*ones(length(-40:1:50),1)',30*ones(length(-40:1:50),1)
    ',60*ones(length(-40:1:50),1)',90*ones(length(-40:1:50),1)'); %
    flexion angle [deg]
zData = cat(1,APForceFitInterpolated0,APForceFitInterpolated30,
    APForceFitInterpolated60,APForceFitInterpolated90); % force [N]

cftool
```

F.2 IE Torque

```
%% Data & Fitting
% 0 deg
%           [Nmm]   [deg]           [deg]
IEdata0 =  [5159.7      -2.742195372    0
            4096.4      -2.826603224    0
            3072.3      -2.855645804    0
            2004.1      -2.87022171     0
            931     -2.864703166    0
            0       -2.864243274    0
            0       -2.84173971     0
            -931        -2.836573846    0
            -2004.1     -2.868639073    0
            -3072.3     -2.871433831    0
            -5159.7     -2.872383463    0];

xIE0 = IEdata0(:,2);
yIE0 = IEdata0(:,3);
zIE0 = IEdata0(:,1);

IETorqueFit0 = fit(xIE0,zIE0,'poly1');

% 30 deg
IEdata30 = [5159.7      9.354652922     30
            3072.3      8.248959791     30
            2004.1      7.394564835     30
            931     5.952171651     30
            0       0.832702691     30
            0       1.568302011     30
            -931        -6.516215436    30
            -2004.1     -8.873982103    30
            -3072.3     -9.877288234    30
            -5159.7     -11.09003142    30];
```

```
xIE30 = IEdata30(:,2);
yIE30 = IEdata30(:,3);
zIE30 = IEdata30(:,1);

IETorqueFit30 = slmengine(xIE30,zIE30,'knots',[-40 40],'increasing','on'
    ,'result','pp');

% 60 deg
IEdata60 = [5159.7  20.83360963     60
            4096.4      19.77855825     60
            3072.3      18.51098364     60
            2004.1      16.76882527     60
            931     13.01570543     60
            0       2.512236383     60
            0       4.219844471     60
            -931        -7.86969913     60
            -2004.1     -12.75853055    60
            -3072.3     -14.60148851    60
            -5159.7     -16.45457736    60];

xIE60 = IEdata60(:,2);
yIE60 = IEdata60(:,3);
zIE60 = IEdata60(:,1);

IETorqueFit60 = slmengine(xIE60,zIE60,'knots',[-40 40],'increasing','on'
    ,'result','pp');

% 90 deg
IEdata90 = [5159.7      24.50676559     90
            3072.3      21.66272683     90
            2004.1      19.51213937     90
            931     14.69069611     90
            0       3.062522188     90
            0       4.779739416     90
            -931        -3.685113407    90
            -2004.1     -8.703705547    90
            -3072.3     -11.88023026    90
            -5159.7     -16.14999389    90];

xIE90 = IEdata90(:,2);
yIE90 = IEdata90(:,3);
zIE90 = IEdata90(:,1);

IETorqueFit90 = slmengine(xIE90,zIE90,'knots',[-40 40],'increasing','on'
    ,'result','pp');

%% Interpolate points from curve fits to be used in surface fit
span = -60:1:60;

IETorqueFitInterpolated0     = feval(IETorqueFit0, span)';
IETorqueFitInterpolated30    = ppval(IETorqueFit30, span);
IETorqueFitInterpolated60    = ppval(IETorqueFit60, span);
IETorqueFitInterpolated90    = ppval(IETorqueFit90, span);

xData = cat(1,span,span,span,span); % rotation [deg]
yData = cat(1,0*ones(length(span),1)',30*ones(length(span),1)',60*ones(
    length(span),1)',90*ones(length(span),1)'); % flexion angle [deg]
zData = cat(1,IETorqueFitInterpolated0,IETorqueFitInterpolated30,
    IETorqueFitInterpolated60,IETorqueFitInterpolated90); % torque [Nm]

cftool
```

G Appendix: MATLAB Code ACL Deficient Data Fit

G.1 AP Force

```
%% Data & Fitting
% 0 deg
APdata0 =  [72.716      6.6 0
            0       -0.3 0
            -20.041     -2.2 0
            -51.597     -3.4 0
            -99.764     -4.5 0];

xAP0 = APdata0(:,2);
yAP0 = APdata0(:,3);
zAP0 = APdata0(:,1);

APForceFit0 = slmengine(xAP0,zAP0,'knots',[min(APdata0(:,2)) max(APdata0
    (:,2))],'increasing','on','result','pp');

% 30 deg
APdata30 = [99.764      10.1 30
            0       1.9 30
            -20.041     -1.5 30
            -51.597     -3.0 30
            -99.764     -4.2 30];

xAP30 = APdata30(:,2);
yAP30 = APdata30(:,3);
zAP30 = APdata30(:,1);

APForceFit30 = slmengine(xAP30,zAP30,'knots',[min(APdata30(:,2)) max(
    APdata30(:,2))],'increasing','on','result','pp');

% 60 deg
APdata60 = [99.764      15.2 60
            0       5.4 60
            -20.041 4.3 60
            -51.597     3.4 60
            -99.764     2.4 60];

xAP60 = APdata60(:,2);
yAP60 = APdata60(:,3);
zAP60 = APdata60(:,1);

APForceFit60 = slmengine(xAP60,zAP60,'knots',[min(APdata60(:,2)) max(
    APdata60(:,2))],'increasing','on','result','pp');

% 90 deg
APdata90 = [99.764      23.5 90
            0       12.8 90
            -20.041     12.1 90
            -51.597     11.4 90
            -99.764     10.5 90];

xAP90 = APdata90(:,2);
yAP90 = APdata90(:,3);
zAP90 = APdata90(:,1);

APForceFit90 = slmengine(xAP90,zAP90,'knots',[min(APdata90(:,2)) max(
    APdata90(:,2))],'increasing','on','result','pp');
```

```
%% Interpolate points from curve fits to be used in surface fit
APForceFitInterpolated0     = ppval(APForceFit0, -40:1:50);
APForceFitInterpolated30    = ppval(APForceFit30, -40:1:50);
APForceFitInterpolated60    = ppval(APForceFit60, -40:1:50);
APForceFitInterpolated90    = ppval(APForceFit90, -40:1:50);

xData = cat(1,-40:1:50,-40:1:50,-40:1:50,-40:1:50); % displacement [mm]
yData = cat(1,0*ones(length(-40:1:50),1)',30*ones(length(-40:1:50),1)
    ',60*ones(length(-40:1:50),1)',90*ones(length(-40:1:50),1)'); %
    flexion angle [deg]
zData = cat(1,APForceFitInterpolated0,APForceFitInterpolated30,
    APForceFitInterpolated60,APForceFitInterpolated90); % force [N]

cftool
```

G.2 IE Torque

```
%% Data & Fitting
% 0 deg
IEdata0 =  [5159.7      -1.6 0
            0       -2.864243274    0
            0       -2.84173971     0
            -931        -2.836573846    0
            -2004.1     -2.868639073    0
            -3072.3     -2.871433831    0
            -5159.7     -2.872383463    0];

xIE0 = IEdata0(:,2);
yIE0 = IEdata0(:,3);
zIE0 = IEdata0(:,1);


% IETorqueFit0 = slmengine(xIE0,zIE0,'knots',[-40 40],'increasing','on
    ','result','pp');

IETorqueFit0 = fit(xIE0,zIE0,'poly1');

% 30 deg
IEdata30 = [5159.7      14.1 30
            0       0.832702691     30
            0       1.568302011     30
            -931        -6.516215436    30
            -2004.1     -8.873982103    30
            -3072.3     -9.877288234    30
            -5159.7     -11.09003142    30];

xIE30 = IEdata30(:,2);
yIE30 = IEdata30(:,3);
zIE30 = IEdata30(:,1);

IETorqueFit30 = slmengine(xIE30,zIE30,'knots',[-40 40],'increasing','on'
    ,'result','pp');

% 60 deg
IEdata60 = [5159.7      23.9    60
            0       2.512236383     60
            0       4.219844471     60
            -931        -7.86969913     60
            -2004.1     -12.75853055    60
            -3072.3     -14.60148851    60
            -5159.7     -16.45457736    60];

xIE60 = IEdata60(:,2);
```

```
yIE60 = IEdata60(:,3);
zIE60 = IEdata60(:,1);

IETorqueFit60 = slmengine(xIE60,zIE60,'knots',[-40 40],'increasing','on'
    ,'result','pp');

% 90 deg
IEdata90 = [5159.7      28.5    90
            0       3.062522188     90
            0       4.779739416     90
            -931        -3.685113407    90
            -2004.1     -8.703705547    90
            -3072.3     -11.88023026    90
            -5159.7     -16.14999389    90];

xIE90 = IEdata90(:,2);
yIE90 = IEdata90(:,3);
zIE90 = IEdata90(:,1);

IETorqueFit90 = slmengine(xIE90,zIE90,'knots',[-40 40],'increasing','on'
    ,'result','pp');

%% Interpolate points from curve fits to be used in surface fit
span = -60:1:60;

IETorqueFitInterpolated0     = feval(IETorqueFit0, span)';
IETorqueFitInterpolated30    = ppval(IETorqueFit30, span);
IETorqueFitInterpolated60    = ppval(IETorqueFit60, span);
IETorqueFitInterpolated90    = ppval(IETorqueFit90, span);

xData = cat(1,span,span,span,span); % rotation [deg]
yData = cat(1,0*ones(length(span),1)',30*ones(length(span),1)',60*ones(
    length(span),1)',90*ones(length(span),1)'); % flexion angle [deg]
zData = cat(1,IETorqueFitInterpolated0,IETorqueFitInterpolated30,
    IETorqueFitInterpolated60,IETorqueFitInterpolated90); % torque [Nm]

cftool
```

H Appendix: MATLAB Code variableStiffnessAP

```
function F = variableStiffnessAP(kneeAngle, u)
    % Function receives current knee angle (kneeAngle) in deg and
        current
    % AP displacement (u) in mm and returns stifness force (F) in N

    % Variable is initiated
    F = zeros(1);

    % Excludes the function APForceFunc from code generation to allow
        Simulink simulation
    coder.extrinsic('APForceFunc');

    % Calls force function. -1 is added so it acts as a spring force
    F = -1*(APForceFunc(kneeAngle, u));
end

function F = APForceFunc(kneeAngle, u)
    % Load laxity fitted data
    S = load('APForceInterpolatedSurfaceFitData');

    % Cast variables to double
    kneeAngle = double(kneeAngle);
    u = double(u);

    % Evaluate the fit
    F = S.APForceInterpolatedSurfaceFit(u,kneeAngle);
end
```

I Appendix: MATLAB Code variableStiffnessIE

```
function T = variableStiffnessIE(kneeAngle, q)
    % Function receives current knee angle (kneeAngle) in deg and
      current
    % IE rotation (q) in rad and returns stifness torque (T) in Nm

    % Variable is initiated
    T = zeros(1);
    % Excludes the function APForceFunc from code generation to allow
      Simulink simulation
    coder.extrinsic('IETorqueFunc');

    % Calls force function. -1 is added so it acts as a spring force
    T = -1*(IETorqueFunc(kneeAngle, q));
end

function T = IETorqueFunc(kneeAngle, q)
    % Conversion from radians to degrees
    w = 180/pi*q;

    % Load laxity fitted data
    S = load('IETorqueInterpolatedSurfaceFitData');

    % Cast variables to double
    kneeAngle = double(kneeAngle);
    w = double(w);

    % Evaluate the fit
    k = S.IETorqueInterpolatedSurfaceFit(w,kneeAngle);
    T = k/(1e+03); % 1e+03 factor for conversion to [Nm]
end
```

J Appendix: MATLAB Code Inverse Plant Fit

J.1 AP Motion

```
%% Data & Fitting
% 0 deg
%            [N]          [mm]     [deg]
APdata0 =  [0    -2.636   -0.461   0
            5    -2.635   -1.059   0
            10   -2.635   -1.466   0
            15   -2.634   -1.786   0
            20   -2.634   -2.053   0
            25   -2.634   -2.284   0
            30   -2.633   -2.490   0
            35   -2.633   -2.676   0
            40   -2.632   -2.847   0
            45   -2.632   -3.004   0
            50   -2.631   -3.151   0
            55   -2.631   -3.289   0
            60   -2.631   -3.419   0
            65   -2.630   -3.543   0
            70   -2.630   -3.660   0
            75   -2.629   -3.772   0
            80   -2.629   -3.879   0];

xAP0 = APdata0(:,3);
yAP0 = APdata0(:,4);
zAP0 = APdata0(:,1);


FFAPForceFit0 = slmengine(xAP0,zAP0,'knots',[-10 20],'decreasing','on','
    result','pp');

% 30 deg
APdata30 = [0    -0.364   0.988    30
            5    -0.360   -0.284   30
            10   -0.357   -0.962   30
            15   -0.353   -1.460   30
            20   -0.350   -1.864   30
            25   -0.346   -2.208   30
            30   -0.343   -2.510   30
            35   -0.340   -2.781   30
            40   -0.336   -3.029   30
            45   -0.333   -3.256   30
            50   -0.330   -3.468   30
            55   -0.327   -3.666   30
            60   -0.323   -3.852   30
            65   -0.320   -4.028   30
            70   -0.317   -4.196   30
            75   -0.314   -4.355   30
            80   -0.311   -4.508   30];

xAP30 = APdata30(:,3);
yAP30 = APdata30(:,4);
zAP30 = APdata30(:,1);

FFAPForceFit30 = slmengine(xAP30,zAP30,'knots',[-10 20],'decreasing','on
    ','result','pp');

% 60 deg
APdata60 = [0    0.630    5.275    60
```

```
            5   0.626   4.834   60
            10  0.621   4.462   60
            15  0.617   4.138   60
            20  0.613   3.849   60
            25  0.608   3.586   60
            30  0.604   3.345   60
            35  0.599   3.121   60
            40  0.595   2.912   60
            45  0.591   2.716   60
            50  0.586   2.530   60
            55  0.582   2.354   60
            60  0.578   2.186   60
            65  0.574   2.025   60
            70  0.569   1.871   60
            75  0.565   1.724   60
            80  0.561   1.581   60];

xAP60 = APdata60(:,3);
yAP60 = APdata60(:,4);
zAP60 = APdata60(:,1);

FFAPForceFit60 = slmengine(xAP60,zAP60,'knots',[-10 20],'decreasing','on
    ','result','pp');

% 90 deg
APdata90 = [0   4.103   12.745  90
            5   4.102   12.436  90
            10  4.102   12.154  90
            15  4.101   11.894  90
            20  4.101   11.653  90
            25  4.100   11.426  90
            30  4.100   11.213  90
            35  4.099   11.011  90
            40  4.098   10.820  90
            45  4.098   10.637  90
            50  4.097   10.462  90
            55  4.097   10.294  90
            60  4.096   10.132  90
            65  4.095   9.976   90
            70  4.095   9.826   90
            75  4.094   9.680   90
            80  4.094   9.539   90];

xAP90 = APdata90(:,3);
yAP90 = APdata90(:,4);
zAP90 = APdata90(:,1);

FFAPForceFit90 = slmengine(xAP90,zAP90,'knots',[-10 20],'decreasing','on
    ','result','pp');


%% Interpolate points from spline fits to be used in surface fit
span = -10:0.1:20;

FFAPForceFitInterpolated0     = ppval(FFAPForceFit0, span);
FFAPForceFitInterpolated30    = ppval(FFAPForceFit30, span);
FFAPForceFitInterpolated60    = ppval(FFAPForceFit60, span);
FFAPForceFitInterpolated90    = ppval(FFAPForceFit90, span);

% displacement [mm]
xData = cat(1,span,span,span,span);
% flexion angle [deg]
```

```
yData = cat(1,0*ones(length(span),1)',30*ones(length(span),1)',60*ones(
    length(span),1)',90*ones(length(span),1)');
% force [N]
zData = cat(1,FFAPForceFitInterpolated0,FFAPForceFitInterpolated30,
    FFAPForceFitInterpolated60,FFAPForceFitInterpolated90);

cftool
```

J.2 IE Motion

```
%% Data & Fitting
% 0 deg
%            IE[deg]          AP[mm]     [N]M(+)  [deg]
IEdata0 =  [-2.635617095        -0.461408067    0       0
            -2.623045764        -0.805853727    5       0
            -2.610464587        -1.073404706    10      0
            -2.597846082        -1.295834141    15      0
            -2.585174073        -1.4880996      20      0
            -2.572437364        -1.658555023    25      0
            -2.559627378        -1.812388575    30      0
            -2.546737049        -1.953065611    35      0
            -2.53376027 -2.083027213    40      0
            -2.520691542        -2.204064293    45      0
            -2.507525773        -2.317533961    50      0
            -2.494258136        -2.424491597    55      0
            -2.480883975        -2.5257764      60      0
            -2.467398733        -2.622066583    65      0
            -2.453797914        -2.713920359    70      0
            -2.440076966        -2.801802107    75      0
            -2.426231395        -2.886103174    80      0
            -2.647667268        -0.787651504    -5      0
            -2.659502931        -1.043607836    -10     0
            -2.671154398        -1.257516328    -15     0
            -2.682640175        -1.443009729    -20     0
            -2.693973221        -1.60781666     -25     0
            -2.705163406        -1.756782962    -30     0
            -2.716218666        -1.893167287    -35     0
            -2.727145626        -2.019275453    -40     0
            -2.737949982        -2.136808072    -45     0
            -2.748636716        -2.247057254    -50     0
            -2.759210264        -2.351035756    -55     0
            -2.769674656        -2.449536168    -60     0
            -2.780033519        -2.54321745     -65     0
            -2.790290224        -2.63261068     -70     0
            -2.800447871        -2.718164764    -75     0
            -2.810509354        -2.80025486     -80     0];

xIE0 = IEdata0(:,1);
yIE0 = IEdata0(:,4);
zIE0 = IEdata0(:,3);


% FFIETorqueFit0 = slmengine(xIE0,zIE0,'knots',[-45 45],'increasing','on
    ','result','pp');
FFIETorqueFit0 = fit(xIE0,zIE0,'poly1');

% 30 deg
IEdata30 = [-0.363955442        0.987778851     0       30
1.387967297     0.208157299     5       30
3.026894385     -0.268572874    10      30
4.245755478     -0.629211178    15      30
5.162066194     -0.925641355    20      30
5.892066936     -1.180293083    25      30
```

```
6.5008113        -1.405173779    30      30
7.025132448      -1.607567116    35      30
7.48738182       -1.792266951    40      30
7.902029988      -1.962616896    45      30
8.27895716       -2.121053028    50      30
8.62520546       -2.26941142     55      30
8.945972605      -2.409114355    60      30
9.245206941      -2.541288531    65      30
9.525980626      -2.666843438    70      30
9.790733065      -2.786525045    75      30
10.04143513      -2.90095366     80      30
-1.717869584     0.209299573     -5      30
-2.711945067     -0.270180521    -10     30
-3.489231729     -0.633947048    -15     30
-4.129483394     -0.933054094    -20     30
-4.676409704     -1.189964626    -25     30
-5.155809408     -1.41679017     -30     30
-5.584034694     -1.620894734    -35     30
-5.972071143     -1.807126405    -40     30
-6.32765062      -1.978867389    -45     30
-6.656423        -2.138581198    -50     30
-6.962645074     -2.288122678    -55     30
-7.249607935     -2.428932432    -60     30
-7.519908699     -2.562145895    -65     30
-7.775636671     -2.68868246     -70     30
-8.018498144     -2.809295814    -75     30
-8.249905908     -2.924612099    -80     30];

xIE30 = IEdata30(:,1);
yIE30 = IEdata30(:,4);
zIE30 = IEdata30(:,3);

FFIETorqueFit30 = slmengine(xIE30,zIE30,'knots',[min(IEdata30(:,1)) max(
    IEdata30(:,1))],'increasing','on','result','pp');

% 60 deg
IEdata60 = [0.629804287 5.27458424       0       60
            4.409590846 5.045724678      5       60
            7.562075251 4.84090527       10      60
            9.637782669 4.653918816      15      60
            11.14682225 4.481093835      20      60
            12.34023937 4.319988081      25      60
            13.33463517 4.168801604      30      60
            14.19186361 4.026155063      35      60
            14.94848248 3.890949175      40      60
            15.62789387 3.762301233      45      60
            16.24600387 3.639484735      50      60
            16.81413515 3.521894502      55      60
            17.34064822 3.409020102      60      60
            17.83190211 3.300426521      65      60
            18.29285324 3.195739723      70      60
            18.72744489 3.094635233      75      60
            19.13886946 2.996829726      80      60
            -2.034342715        5.044501721      -5      60
            -3.850427719        4.837276256      -10     60
            -5.227144119        4.647775853      -15     60
            -6.345352117        4.472603385      -20     60
            -7.293664058        4.309331957      -25     60
            -8.121420199        4.156143666      -30     60
            -8.858858253        4.011627733      -35     60
            -9.525870363        3.874678019      -40     60
            -10.1362645 3.744390797      -45     60
            -10.70003493        3.620028025      -50     60
            -11.2246616 3.50096886      -55     60
            -11.71589757        3.386701159     -60     60
            -12.17826923        3.276776112     -65     60
            -12.61540658        3.170814627     -70     60
            -13.03026867        3.068489008     -75     60
            -13.42530195        2.96950579      -80     60];

xIE60 = IEdata60(:,1);
yIE60 = IEdata60(:,4);
zIE60 = IEdata60(:,3);

FFIETorqueFit60 = slmengine(xIE60,zIE60,'knots',[min(IEdata60(:,1)) max(
    IEdata60(:,1))],'increasing','on','result','pp');

% 90 deg
IEdata90 = [4.102542368 12.74480723     0       90
            7.005341939 12.5862409      5       90
            9.758312942 12.43529489     10      90
            11.99990554 12.29115623     15      90
            13.78197561 12.15313679     20      90
            15.23725075 12.02064427     25      90
            16.46329442 11.89316584     30      90
            17.52313931 11.77026301     35      90
            18.45787975 11.65155225     40      90
            19.29527459 11.53669912     45      90
            20.05479575 11.42540974     50      90
            20.75057468 11.31742426     55      90
            21.39317361 11.21251187     60      90
            21.99068881 11.1104663      65      90
            22.5494621  11.01110273     70      90
            23.07455458 10.91425451     75      90
            23.57007114 10.819771       80      90
            1.59342886  12.58639198     -5      90
            -0.415232449        12.43557955     -10     90
            -2.043239067        12.29156455     -15     90
            -3.403989627        12.15360867     -20     90
            -4.573      12.02113332     -25     90
            -5.601      11.89363526     -30     90
            -6.519      11.77067108     -35     90
            -7.350233494        11.65187034     -40     90
            -8.111420404        11.53689883     -45     90
            -8.81434709 11.42546465     -50     90
            -9.468149682        11.3173033      -55     90
            -10.07992736        11.21220509     -60     90
            -10.65530664        11.10994728     -65     90
            -11.19882393        11.01035167     -70     90
            -11.71419159        10.913253       -75     90
            -12.20448772        10.81850586     -80     90];

xIE90 = IEdata90(:,1);
yIE90 = IEdata90(:,4);
zIE90 = IEdata90(:,3);

FFIETorqueFit90 = slmengine(xIE90,zIE90,'knots',[min(IEdata90(:,1)) max(
    IEdata90(:,1))],'increasing','on','result','pp');

%% Interpolate points from spline fits to be used in surface fit
span = -90:1:90;

FFIETorqueFitInterpolated0     = feval(FFIETorqueFit0, span)';
FFIETorqueFitInterpolated30    = ppval(FFIETorqueFit30, span);
FFIETorqueFitInterpolated60    = ppval(FFIETorqueFit60, span);
FFIETorqueFitInterpolated90    = ppval(FFIETorqueFit90, span);
```

```
xData = cat(1,span,span,span,span); % rotation [deg]
yData = cat(1,0*ones(length(span),1)',30*ones(length(span),1)',60*ones(
    length(span),1)',90*ones(length(span),1)'); % flexion angle [deg]
zData = cat(1,FFIETorqueFitInterpolated0,FFIETorqueFitInterpolated30,
    FFIETorqueFitInterpolated60,FFIETorqueFitInterpolated90); % Force [N
    ] M(+)

cftool
```

Bibliography

Alici, G. (2018), Softer is Harder: What Differentiates Soft Robotics from Hard Robotics, *MRS Advances*, **vol. 3 28**, p. 1557–1568, doi:10.1557/adv.2018.159.

Beynnon, B. D., B. C. Fleming, D. L. Churchill and D. Brown (2003), The Effect of Anterior Cruciate Ligament Deficiency and Functional Bracing on Translation of the Tibia Relative to the Femur during Nonweightbearing and Weightbearing, *The American Journal of Sports Medicine*, **vol. 31 1**, p. 99–105, doi:10.1177/03635465030310012801.

Beynnon, B. D., M. H. Pope, C. M. Wertheimer, R. J. Johnson, B. C. Fleming, C. E. Nichols and J. G. Howe (1992), The effect of functional knee-braces on strain on the anterior cruciate ligament in vivo., *The Journal of Bone & Joint Surgery*, **vol. 74 9**, p. 1298–1312, doi:10.2106/00004623-199274090-00003.

Bovi, G., M. Rabuffetti, P. Mazzoleni and M. Ferrarin (2011), A multiple-task gait analysis approach: Kinematic, kinetic and EMG reference data for healthy young and adult subjects, *Gait & Posture*, **vol. 33 1**, p. 6–13, doi:10.1016/j.gaitpost.2010.08.009.

Butler, D. L., F. R. Noyes and E. S. Grood (1980), Ligamentous restraints to anterior-posterior drawer in the human knee. A biomechanical study., *The Journal of Bone & Joint Surgery*, **vol. 62 2**, p. 259–270, doi:10.2106/00004623-198062020-00013.

Chachula, L., K. L. Cameron, S. J. Svoboda and B. D. Owens (2012), Postop bracing after ACL reconstruction, *Lower Extremity Review Magazine*, accessed: 13-07-2020 https://lermagazine.com/article/postop-bracing-after-acl-reconstruction.

Daerden, F. and D. Lefeber (2002), Pneumatic Artificial Muscles: actuators for robotics and automation, *European Journal of Mechanical and Environmental Engineering*, **vol. 47**.

D'Errico, J. (2017), Slm - Shape Language Modeling MATLAB Toolbox, accessed: 25-02-2021 https://www.mathworks.com/matlabcentral/fileexchange/24443-slm-shape-language-modeling.

Dhillon, M. S., K. Bali and S. Prabhakar (2011), Proprioception in anterior cruciate ligament deficient knees and its relevance in anterior cruciate ligament reconstruction, *Indian Journal of Orthopaedics*, **vol. 45 4**, pp. 294–300, doi:10.4103/0019-5413.80320.

ExRx.net (2021), Body Segment Data, accessed: 20-05-2021 https://exrx.net/Kinesiology/Segments.

Fox, A. J., A. Bedi and S. A. Rodeo (2011), The Basic Science of Human Knee Menisci, *Sports Health: A Multidisciplinary Approach*, **vol. 4 4**, p. 340–351, doi:10.1177/1941738111429419.

Ganesh, M. (2020a), *An improved soft-robotic knee brace to enhance the knee stability for ACL deficient patients*, Master's thesis, University of Twente, the Netherlands.

Ganesh, M. (2020b), Force Characterization of Pneumatic Artifcial Muscles for Improved Soft Robotic Knee Brace, capita Selecta Report, University of Twente, the Netherlands.

Hoffman, M. (2019), Knee (Human Anatomy): Function, Parts, Conditions, Treatments, accessed: 22-02-2021 https://www.webmd.com/pain-management/knee-pain/picture-of-the-knee#1.

Joint Health Matters (2017), Joint Health Matters: Muscle/Tendon Injury, accessed: 22-02-2021 https://jointhealthmatters.org/knee-injury/muscletendon-injury/.

Khambati, H. (2019), A soft robotic knee brace for improving stability in anterior cruciate ligament deficient patients, Master Thesis, University of Twente.

Lu, T.-W., H.-C. Lin and H.-C. Hsu (2006), Influence of functional bracing on the kinetics of anterior cruciate ligament-injured knees during level walking, *Clinical Biomechanics*, **vol. 21 5**, p. 517–524, doi:10.1016/j.clinbiomech.2005.12.017.

Macaulay, A. A., D. C. Perfetti and W. N. Levine (2011), Anterior Cruciate Ligament Graft Choices, *Sports Health: A Multidisciplinary Approach*, **vol. 4 1**, p. 63–68, doi:10.1177/1941738111409890.

Mader, K., D. Pennig, J. Dargel, M. Gotter, J. Koebke and R. Schmidt-Wiethoff (2007), Biomechanics of the anterior cruciate ligament and implications for surgical reconstruction, *Strategies in Trauma and Limb Reconstruction*, **vol. 2 1**, p. 1–12, doi:10.1007/s11751-007-0016-6.

Majewski, M., H. Susanne and S. Klaus (2006), Epidemiology of athletic knee injuries: A 10-year study, *The Knee*, **vol. 13 3**, p. 184–188, doi:10.1016/j.knee.2006.01.005.

Mannel, H., F. Marin, L. Claes and L. Dürselen (2004), Anterior cruciate ligament rupture translates the axes of motion within the knee, *Clinical Biomechanics*, **vol. 19 2**, p. 130–135, doi:10.1016/j.clinbiomech.2003.11.007.

Masouros, S., A. Bull and A. Amis (2010), (i) Biomechanics of the knee joint, *Orthopaedics and Trauma*, **vol. 24 2**, p. 84–91, doi:10.1016/j.mporth.2010.03.005.

Mayr, H. O., A. Hochrein, W. Hein, R. Hube and A. Bernstein (2010), Rehabilitation results following anterior cruciate ligament reconstruction using a hard brace compared to a fluid-filled soft brace, *The Knee*, **vol. 17 2**, p. 119–126, doi:10.1016/j.knee.2009.07.002.

Moore, K. L. (1993), *Iterative learning control for deterministic systems*, Springer.

Naghibi Beidokhti, H., D. Janssen, S. van de Groes, J. Hazrati, T. Van den Boogaard and N. Verdonschot (2017a), The influence of ligament modelling strategies on the predictive capability of finite element models of the human knee joint, *Journal of Biomechanics*, **vol. 65**, p. 1–11, doi:10.1016/j.jbiomech.2017.08.030.

Naghibi Beidokhti, H., D. Janssen, S. van de Groes and N. Verdonschot (2017b), The peripheral soft tissues should not be ignored in the finite element models of the human knee joint, *Medical & Biological Engineering & Computing*, **vol. 56**, p. 1189–1199, doi:10.1007/s11517-017-1757-0.

Naghibi Beidokhti, H., D. Janssen, T. Van Tienen, S. Van de Groes, T. Van de Boogaard and N. Verdonschot (2020), A novel approach for optimal graft positioning and tensioning in anterior cruciate ligament reconstructive surgery based on the finite element modeling technique, *The Knee*, **vol. 27**, p. 384–396, doi:10.1016/j.knee.2020.01.010.

Neuman, P., I. Kostogiannis, T. Fridén, H. Roos, L. E. Dahlberg and M. Englund (2010), Knee laxity after complete anterior cruciate ligament tear: a prospective study over 15 years, *Scandinavian Journal of Medicine & Science in Sports*, **vol. 22 2**, p. 156–163, doi:10.1111/j.1600-0838.2010.01157.x.

Obbink, R. O. (2019), Learning feed-forward control for a Soft Robotics phantom to simulate respiratory liver motion, report, University of Twente, the Netherlands.

OrthoInfo (2014), Anterior Cruciate Ligament (ACL) Injuries - OrthoInfo - AAOS, accessed: 17-02-2021 https://orthoinfo.aaos.org/en/diseases--conditions/anterior-cruciate-ligament-acl-injuries/.

OrthoLoad (2021), Orthoload: Loading of Orthopaedic Implants, accessed: 27-04-2021 https://orthoload.com/.

Papadonikolakis, A., L. Cooper, N. Stergiou, A. D. Georgoulis and P. N. Soucacos (2003), Compensatory mechanisms in anterior cruciate ligament deficiency, *Knee Surgery, Sports Traumatology, Arthroscopy*, **vol. 11 4**, p. 235–243, doi:10.1007/s00167-003-0367-6.

Paterno, M. V. (2017), Non-operative Care of the Patient with an ACL-Deficient Knee, *Current Reviews in Musculoskeletal Medicine*, **vol. 10 3**, p. 322–327, doi:10.1007/s12178-017-9431-6.

Paterno, M. V., M. J. Rauh, L. C. Schmitt, K. R. Ford and T. E. Hewett (2014), Incidence of Second ACL Injuries 2 Years After Primary ACL Reconstruction and Return to Sport, *The American Journal of Sports Medicine*, **vol. 42 7**, p. 1567–1573, doi:10.1177/0363546514530088.

Potter, H. G., S. K. Jain, Y. Ma, B. R. Black, S. Fung and S. Lyman (2011), Cartilage Injury After Acute, Isolated Anterior Cruciate Ligament Tear, *The American Journal of Sports Medicine*, **vol. 40 2**, p. 276–285, doi:10.1177/0363546511423380.

Shelburne, K. B., M. R. Torry and M. G. Pandy (2005), Effect of Muscle Compensation on Knee Instability during ACL-Deficient Gait, *Medicine & Science in Sports & Exercise*, **vol. 37 4**, p. 642–648, doi:10.1249/01.mss.0000158187.79100.48.

Shenoy, R., P. Pastides and D. Nathwani (2013), (iii) Biomechanics of the knee and TKR, *Orthopaedics and Trauma*, **vol. 27 6**, p. 364–371, doi:10.1016/j.mporth.2013.10.003.

Vassalou, E. E., M. E. Klontzas, G. K. Kouvidis, P. I. Matalliotaki and A. H. Karantanas (2016), Rotational Knee Laxity in Anterior Cruciate Ligament Deficiency: An Additional Secondary Sign on MRI, *American Journal of Roentgenology*, **vol. 206 1**, p. 151–154, doi:10.2214/ajr.15.14816.

Williams, G. N., P. J. Barrance, L. Snyder-Mackler and T. S. Buchhanan (2004), Altered Quadriceps Control in People with Anterior Cruciate Ligament Deficiency, *Medicine & Science in Sports & Exercise*, **vol. 36 7**, p. 1089–1097, doi:10.1249/01.mss.0000131959.20666.11.

Williams, G. N., L. Snyder-Mackler, P. J. Barrance and T. S. Buchanan (2005), Quadriceps femoris muscle morphology and function after ACL injury: a differential response in copers versus non-copers, *Journal of Biomechanics*, **vol. 38 4**, p. 685–693, doi:10.1016/j.jbiomech.2004.04.004.

Wilson, A. (2018), Knee anatomy including ligaments, cartilage and meniscus, accessed: 17-02-2021 https://profadrianwilson.co.uk/knee-treatments/knee-anatomy/.