Designing a wearable sensor system to monitor the internal mechanism of a stance phase control passive prosthetic knee

by

Rachel Reding

A thesis submitted in conformity with the requirements for the degree of Master of Health Science, Clinical Engineering
Institute of Biomaterials and Biomedical Engineering
University of Toronto

© Copyright by Rachel Reding 2018
Designing a wearable sensor system to monitor the internal mechanism of a stance phase control passive prosthetic knee

Rachel Reding
Master of Health Science, Clinical Engineering
Institute of Biomaterials and Biomedical Engineering
University of Toronto
2018

Abstract

Prosthetic knees aim to improve mobility for above-knee amputees, but studies that quantify the performance of prosthetic knee controllers are limited. Further, most studies about the performance of prosthetic knees are often limited to a controlled environment, so information from gait studies that inform design decisions may be misleading. Therefore, the objective of this project was to further develop a wearable gait analysis tool that quantitatively measures performance of a prosthetic knee.

The idea of collecting empirical data and inputting it into a computational model for design improvements was motivated through a literature review. Moreover, a fully wearable gait analysis tool and 4 finite output states were developed. Both were validated through collection of data from walking trials with able-bodied participants.

The further development of this tool has shown the potential for a wearable gait analysis tool that can be used in a variety of environmental conditions, to inform design decisions.
Acknowledgments

I would like to acknowledge my thesis supervisor Dr. Jan Andrysek for his guidance during this process, and for allowing me to be part of a technology that will have an impact on so many others. I would also like to thank my committee members, Dr. José Zariffa and Dr. Kei Masani, for their insight and support. I also had the fortune of working with two great postdoctoral researchers, Dr. Matt Leineweber and Dr. Arezoo Eshraghi: thank you for answering my many questions with such patience. Further, I would like to thank the staff at Holland Bloorview who provided guidance and support during my Master’s.

To the members of Propel whom I had the pleasure of sharing our workspace with: Sam, Mark, Brock, Rafael, Megan, Emerson, thank you for supporting me, helping me, and laughing with me. To Calvin, you taught me so much, I am grateful to have been your neighbor in the lab. Maggie, Sam and Victoria, thank you for your contributions to this study. Jessica, your contributions and kind words are forever invaluable to me.

To my classmates, thank you for being interested in my project and ambitions. You pushed me to be a better academic and enriched my graduate experience. To my friends, both in Toronto and around the world, I could not have done this without you. To my friends in the Prism Lab, thank you for inviting me to lunch and much needed coffee breaks. Rami and Arushri, thank you for your never ending positivity and words of wisdom.

And finally, to all my loved ones, you have been nothing but amazing during this entire process. Kyle, thank you for being wholly invested in my passions and pursuit of knowledge, and for having unwavering confidence in me. Jason, Sarah and Josh, you have been the most supportive siblings. Mom and Dad, thank you for always taking my long phone calls, for the visits, and for having such an interest in my passions. To Mom, thank you for forcing me to always practice my presentations more than I think I should! To Lily, last but not least, thank you for being the one to stay with me during the ups and downs. And thank you for sitting on my laptop in the most intense of times, giving me a laugh when I needed it most.
# Table of Contents

Acknowledgments .................................................................................................................. iii

Table of Contents ................................................................................................................... iv

List of Tables ............................................................................................................................. vii

List of Figures ............................................................................................................................ x

List of Appendices ..................................................................................................................... xiv

List of Equations ....................................................................................................................... xv

Abbreviations ............................................................................................................................. xvi

Chapter 1 ...................................................................................................................................... 1

1 Introduction ............................................................................................................................... 1

1.1 Thesis Road Map ................................................................................................................... 1

1.2 Background ........................................................................................................................... 2

1.2.1 Lower Limb Amputation .................................................................................................. 2

1.2.2 Overview of Prosthetic Knee Technologies ..................................................................... 4

Chapter 2 ...................................................................................................................................... 9

2 Background Literature ............................................................................................................. 9

2.1 Gait Cycle ............................................................................................................................. 9

2.2 Studying Prosthesis Function ............................................................................................... 10

2.2.1 Examples of Gait Analysis Techniques .......................................................................... 10

2.2.2 Computational Modeling ................................................................................................. 11

2.3 Characterizing Stance Control Function of AT-Knee .......................................................... 12

2.3.1 Modeling Method ............................................................................................................. 12

2.3.2 Overview of ASPL-SS ..................................................................................................... 13

2.3.3 Data Analysis ................................................................................................................... 15

2.3.4 Results of Characterizing Stance Control ...................................................................... 15

Chapter 3 .................................................................................................................................... 17
6.2 Results ................................................................................................................................. 49
  6.2.1 Comparison of Extension and Flexion Detection ......................................................... 49
  6.2.2 State Analysis ............................................................................................................... 53
  6.3 Discussion ....................................................................................................................... 61
    6.3.1 Measuring WGAT Function ...................................................................................... 61
Chapter 7 ..................................................................................................................................... 65
  7 Limitations and Future Work ............................................................................................. 65
Chapter 8 ..................................................................................................................................... 69
  8 Conclusion .......................................................................................................................... 69
References .................................................................................................................................... 70
Appendices ............................................................................................................................... 79
List of Tables

Table 1: Possible states of the three input variables, and the resulting control state [34]........... 15

Table 2: Design requirements for the technical development of the wearable gait analysis tool. 19

Table 3: Results of literature review. Each paper is classified by the phases of the design cycle discussed in the paper, indicated by a checkmark. The papers are sorted chronologically. ....... 22

Table 4: Summary of benefits and shortcomings of Prototype 1.................................................. 30

Table 5: Overview of features in Prototype 2 that address the shortcomings of Prototype 1..... 32

Table 6: Summary of sensors, what they measure, the purpose of the sensors and the conventions of the measurements........................................................................................................ 43

Table 7: 4 finite control states, based on the binary detection of extension and flexion by the FSR and LPS........................................................................................................................................ 49

Table 8: Thresholds for determination of extension and flexion for FSR, angled LPS and lock position. Further, synchronization type for each participant is displayed................................. 49

Table 9: Accuracy of the angled LPS in detecting extension/flexion compared to the KA moment, for all participants................................................................. 51

Table 10: Accuracy of the force-sensing resistor in detecting extension/flexion compared to the KA moment, for all participants................................................................. 52

Table 11: Accuracy of the LPS in detecting extension/flexion compared to the CA moment, for all participants............................................................................................................ 53

Table 12: Conditions of 4 finite output states compared to 8-state model, based on using CA moment as determination of expected and unexpected states........................................ 54

Table 13: Condition 1: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 3, B and 7, C and 2, and D and 6. Statistically significant differences ( p < 0.05) are indicated by *.......................................................... 55
Table 14: Condition 2: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 3+4, B and 7+8, C and 1+2, and D and 5+6. Statistically significant differences (p < 0.05) are indicated by *.

Table 15: Condition 3: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 1+3, B and 5+7, C and 2+4, and D and 6+8.

Table 16: Results from repeated measures ANOVA. The first column indicates the state of the 4 finite output states used in the calculation, and the second column indicates the state or combination of states from the 8-state model. The mean differences and p-values are in the third and fourth columns, respectively. Statistically significant differences (p < 0.05) are indicated by *.

Table 17: Results from t-tests comparing all possible combinations of the three conditions that correspond to state A, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

Table 18: Results from t-tests comparing all possible combinations of the three conditions that correspond to state B, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

Table 19: Results from t-tests comparing all possible combinations of the three conditions that correspond to state C, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

Table 20: Results from t-tests comparing all possible combinations of the three conditions that correspond to state D, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.
Table 21: Average experimental results from the three trials of the distance test.

Table 22: Percentage of stance phase for states 2, 3, 6, and 7, and 2, 3, 4, 6, 7 and 8 of the 8-state model.

Table 23: Percentage of stance phase spent in extension and flexion as measured by the load transducer and the WGAT.

Table 24: Mean difference in percentage of gait cycle spent in flexion, as calculated from data collected by the load transducer subtracted from either the angled LPS or FSR. SD is the standard deviation. Positive values indicate over-detection of flexion by the sensor as compared to the load transducer.

Table 25: Angled LPS sensitivity analysis. Maximum accuracy across all participants was for a threshold of -0.19 mm, yielding an accuracy of 78.2%.

Table 26: FSR sensitivity analysis. Maximum accuracy across all participants was for a threshold of 380 bits, yielding an accuracy of 83.4%.
List of Figures

Figure 1: Unilateral transfemoral amputee with an amputation of the right leg [7]. ....................... 3

Figure 2: The All-Terrain Knee: (A) Picture of prosthesis assembly; (B) Labeled diagram of components; (C) Disengaged knee lock; and (D) Knee flexion of approximately 120 degrees [23]. ................................................................................................................................. 7

Figure 3: Schematic of the AT-Knee during a typical gait cycle. The knee lock is engaged at the far left, preventing flexion in stance phase so that weight can be loaded. Then the knee lock begins to disengage as the gait cycle progresses to toe off, and the knee lock is completely disengaged through swing phase. Finally, on the right the knee is locked again as the leg has swung forward and commenced a new gait cycle with heel strike................................................................. 7

Figure 4: A typical gait cycle [27]. Stance phase and swing phase are labeled for one gait cycle of the right leg. Stance phase occurs when the leg transitions through weight bearing of heel strike to toe off, and then swing phase occurs as the leg swings forward to make contact with the ground again at the subsequent heel strike................................................................................... 9

Figure 5: Labeled AT-Knee with variables used in the computational model. Both the $M_{CA}$ and $M_{KA}$ show an extension moment [34]. .......................................................................................................................... 13

Figure 6: Picture of ASPL-SS sensor placement and mounting. (A) Inductive proximity sensor, (B) force-sensing resistor, (C) accelerometer, and (D) load transducer [26]......................... 14

Figure 7: Expected states of the 8-state model, and the corresponding stage in stance phase.
Adapted from [34].......................................................................................................................... 16

Figure 8: Iterative design cycle, modified from [36]–[38]......................................................... 22

Figure 9: Placement of the LPS [68], perpendicular to the knee lock. Image of the LPS is enlarged for visibility. .......................................................... 27

Figure 10: WGAT sensors and data logging. .................................................................................... 29

Figure 11: First prototype of the sensor mount and electronics containment. All electronics and sensors were housed in a single, 3D-printed container. (left) Front view. (right) Side view...... 31
Figure 12: Schematic of the second prototype of the sensor mount. (A) *Front view.* Arrow indicates the where the accelerometer attaches to the mount via screws. (B) *Side view.* Top arrow indicates the channel to hold a single cord with the wires that leads to the box on the thigh. The bottom arrow indicates the pocket for the LPS that measures lock position. (C) *Side back view.* The top arrow shows the curved design of the mount that fits with the AT-Knee cover, and the attachment mechanism via screws. The bottom arrow indicates the pocket for the angled LPS.

Figure 13: Setup of second prototype on an able-bodied participant. The second prototype included a 3D-printed mount for the sensors, and an off-the-shelf plastic box for the associated electronics, which was secured to the thigh with Velcro straps.

Figure 14: Setup of the flat block in the vise, with the sensor secured in the chuck of the milling machine [69].

Figure 15: Results from the calibration of the linear position sensor for the distance test.

Figure 16: Results from the calibration of the linear position sensor for the hysteresis test.

Figure 17: Lock position as measured by the linear position sensor and inductive proximity sensor for simulated normal walking.

Figure 18: Lock position as measured by the linear position sensor and inductive proximity sensor for simulated normal walking with heavy heel strike.

Figure 19: Lock position as measured by the linear position sensor and inductive proximity sensor for simulated running.

Figure 20: Lock position as measured by the linear position sensor and inductive proximity sensor for simulated tiptoeing.

Figure 21: All-Terrain Knee instrumented with ATI Mini58 F/T Transducer using custom adaptor plates above and below the transducer. *(left)* Front view. *(right)* Side view [26].

Figure 22: Schematic illustrating the offsets between the ATI Mini58 F/T Transducer Y-axis (red), the ASPL control axis (pink), and the ASPL knee axis (maroon) [26].
Figure 23: Prosthetic gait simulator, used by able-bodied participants. Image on the right shows the inserts designed to decrease the diameter of the gait simulator socket. ......................................... 45

Figure 24: Example data for comparison of KA moment to angled LPS. Figure includes measurement of KA moment (units: Nm; convention: extension moment positive) and angled LPS (units: mm X 5; convention: flexion negative). Each of the 4 states (correct detection of extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (left) Participant 1AB and (right) Participant 3AB............................................. 50

Figure 25: Example data for comparison of KA moment to the force-sensing resistor. Figure includes measurement of KA moment (units: Nm; convention: extension moment positive) and force (units: bits/50; convention: extension positive). Each of the 4 states (correct detection of extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (left) Participant 1AB and (right) Participant 3AB............................................. 51

Figure 26: Example of data for comparison of CA moment to lock position. Figure includes measurement of CA moment (units: Nm; convention: extension moment positive) and LPS (units: mm X 10; convention: locked < threshold, unlocked > threshold). Each of the 4 states (correct detection of extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (left) Participant 1AB and (right) Participant 4AB....... 53

Figure 27: State analysis using the load transducer compared to the WGAT during stance phase. Gait phase is delineated by the vertical blue lines, and states are depicted as on (high) or off (low). (left) Participant 5 and (right) Participant 6. ......................................................................................... 55

Figure 28: Plots of the mean percentage of stance phase for each state. (A) Plots of state A and states 3, 3+4 and 1+3. (B) Plots of state B and states 7, 7+8 and 5+7. (C) Plots of state C and states 2, 1+2 and 2+4. (D) Plots of state D and states 6, 5+6 and 6+8................................................. 57

Figure 29: Plots of mean difference between the 4 finite output states and the corresponding condition. Statistically significant differences from the post-hoc paired t-tests ( p < 0.05) are indicated by *.................................................................. 58

Figure 30: Three low-fidelity prototypes used for the design critique and cognitive walkthrough. Some of the distinct features were: (left) using a linear layout, having a separate button for
recording, and using a checkmark for calibration; (middle) using a linear layout, and using a “refresh” symbol for calibration; and (right) using a single Red-Green-Blue (RGB) LED to communicate all functions.

Figure 31: Two higher-fidelity prototypes. Some of the distinct features were: (left) symbols directly on buttons, and sequential symbols directly on the interface; (right) symbols directly on the interface, user actions and indicators separated by solid line.

Figure 32: CAD diagrams of final prototype. (left) Angled view of the front face. (right) Front view.
List of Appendices

Appendix A: Arduino Code .................................................................................................................. 79

Appendix B: MATLAB Code for Extension/Flexion Detection Comparison ............................... 81

Appendix C: MATLAB Code for State Analysis ............................................................................... 91

Appendix D: Design of User Interface ............................................................................................ 104

Appendix E: Technical Documentation for Linear Position Sensor ............................................. 108

Appendix F: Results from Calibration of LPS ................................................................................. 111

Appendix G: Percentage of Stance Phase for 8-State Model ......................................................... 112

Appendix H: Analysis of Extension and Flexion of WGAT Compared to Load Cell ............... 113

Appendix I: Sensitivity Analysis ...................................................................................................... 114
List of Equations

\[ MCA = MCALc - F1 \times \Delta d1 + F2 \times \Delta d2 \] .................................................. 12

\[ MKA = MKALc + F2 \times \Delta d3 \] .................................................. 12

\[ MCALc = TY + FX \times \Delta z1 + FZ \times \Delta x1 \] .................................................. 13

\[ MKALc = TY + FX \times \Delta z2 + FZ \times \Delta x2 \] .................................................. 13

\[ y = 80.55x + 4 \times 10^{-13} \] .................................................. 35

Control Axis Moment = TY + FX*\Delta z1 + FZ*\Delta x1 .................................................. 41

Knee Axis Moment = TY + FX*\Delta z2 + FZ*\Delta x2 .................................................. 41

\[ \text{accuracy} = \text{number of events of correct extension} + \]
\[ \text{number events of correct flexion} \times \text{total events} \times 100 \] .................................................. 47
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB</td>
<td>Able-bodied</td>
</tr>
<tr>
<td>ACC</td>
<td>Accelerometer</td>
</tr>
<tr>
<td>ACCx</td>
<td>Horizontal acceleration</td>
</tr>
<tr>
<td>ACCz</td>
<td>Vertical acceleration</td>
</tr>
<tr>
<td>AK</td>
<td>Above-knee</td>
</tr>
<tr>
<td>ASPL</td>
<td>Automatic Stance phase Lock</td>
</tr>
<tr>
<td>ASPL-SS</td>
<td>Automatic Stance phase Lock-Sensing System</td>
</tr>
<tr>
<td>AT</td>
<td>All-Terrain</td>
</tr>
<tr>
<td>FSR</td>
<td>Force-sensing resistor</td>
</tr>
<tr>
<td>FZ</td>
<td>Vertical force</td>
</tr>
<tr>
<td>IPS</td>
<td>Inductive proximity sensor</td>
</tr>
<tr>
<td>LPS</td>
<td>Linear position sensor</td>
</tr>
<tr>
<td>WGAT</td>
<td>Wearable gait analysis tool</td>
</tr>
</tbody>
</table>
Chapter 1

1 Introduction

1.1 Thesis Road Map

The purpose of this project was to increase the portability of a wearable gait analysis tool, and define a new way to quantify the performance of a stance phase controller through the use of 4 finite output states determined via the sensor outputs of the wearable gait analysis tool. The definition of 4 finite output states was required because of the new, portable sensors used in the data collection instrumentation. The 4 finite outputs could be inputted into a computational model to characterize the performance and inform design decisions of prosthetic knees, by perturbing known variables and seeing if the result matches what would be expected.

The following document has been divided into 8 chapters. Chapter 1 starts by providing an overview of lower limb amputation and then gives a review of some of the prosthetic technologies available. Further, it details the development of the All-Terrain Knee and Automatic Stance Phase Lock mechanism.

Chapter 2 is a survey of the relevant literature to the research question.

Chapter 3 states the research problem and objectives by outlining specific goals and design requirements.

Chapter 4 details the technical literature review. 4.1 provides the detailed methodology, 4.2 provides the results and 4.3 discusses the results and how they motivate the technical work of this project.

Chapter 5 is an overview of the design and preliminary benchtop tests. 5Sections 5.1 and 6.1.2 describe the development of the Wearable Gait Analysis Tool (WGAT), and the instrumentation and experimental protocols used to collect data for its evaluation.

Chapter 6 discusses the measurement of the WGAT function, as well as the determination of the 4 output states. Section explains the data processing done to evaluate the performance of the WGAT.
Chapter 7 explains some of the projects limitations, and potential implications for future work that are worth exploring.

To conclude, chapter 8 restates the research problem and summarizes the main findings of this research and its significance.

1.2 Background

1.2.1 Lower Limb Amputation

An estimated one in 150 North Americans are currently living with an amputation, and the number of people living with an amputation is expected to reach 3.6 million by 2050 in the United States alone [1]. Studies suggest that approximately 65% of amputations involve the lower limb, of which about 30% are accounted for by transfemoral (above-knee) amputations [1]. Causes of amputation include diabetes mellitus, dysvascular disease, trauma, and malignancy of the bone and joint [2]. The prevalence of diabetes in the United States is projected to nearly double by the year 2030, and given the increase in the prevalence of obesity and the known relationship between obesity and diabetes, a projected increase in the incidence of amputations secondary to dysvascular conditions is likely [1], [3].

Amputation is often regarded as the final failure of orthopaedic or vascular treatment, and lower limb amputations specifically constitute a major physical disability with life-long physical challenges [4], [5]. However, major benefits of amputation are that it can often eliminate a painful limb ultimately bringing relief to the patient and it can also allow rehabilitation of the patient to the status of a functional ambulatory individual [4], especially since modern treatments have drastically reduced the mobility limitations associated with a missing limb [5].

1.2.1.1 Transfemoral Amputation and Mobility Function

As mentioned previously, transfemoral amputation is a lower limb amputation above-the-knee (see Figure 1). Transfemoral amputees often experience a decrease in muscle strength, which results from several factors including reduction in muscle mass after amputation, inadequate mechanical fixation of muscles, and atrophy of the remaining musculature, which can be attributed to the fact that sectioning of the thigh muscles removes muscle bulk and interferes with nerve innervation to the sectioned muscle [6]. Transfemoral amputees often have a
shortened effective moment arm, so surgeons usually aim to preserve the length of the residual limb as much as possible, and maintain the femur in a central position in the soft tissue to minimize the adverse effect on the amputee’s gait [6].

Figure 1: Unilateral transfemoral amputee with an amputation of the right leg [7].

However, transfemoral amputees often experience mobility limitations including falling, fear of falling, higher energy consumption during gait, and reduced walking speeds [6]. Firstly, falling is a health risk, that can result in serious injury or death [8]–[10]. In a study that received questionnaire responses from 396 participants, Gauthier-Gagnon et al found that 50% of amputee respondents reported at least one fall in the past month, and a significantly higher proportion of transfemoral amputees (63.9%) fell as compared to transtibial amputees [11]. Difficulties usually concern activities performed in an open environment in which the individual must predict and adapt rapidly to environmental changes or activities requiring greater stability and motor control because the center of mass is critically moved over and out of the base of support [11].
Specifically, amputees have identified the following activities as higher falling risk: picking up and carrying objects, ascending/descending stairs, and walking in inclement weather [11].

Secondly, in addition to falling is the fear of falling. In the same study by Gauthier-Gagnon et al, 50.8% of transtibial and transfemoral amputees had to think of every step they made with their prosthesis, which reduces the automaticity and confidence in gait [11]. Further, 23.5% could not walk more than 30 steps at a time, one of the reasons being that respondents stated they were afraid of falling [11]. More specifically, in another study, Miller et al showed that 49.2% of respondents had a fear of falling [12]. In response to falling and the fear of falling, amputees have reported requiring the use of walking aids (i.e., canes and walkers), which can decrease their independence and autonomy [11]. Moreover, in the study by Miller et al, 76.2% of respondents reported avoiding everyday activities because of this fear [12], and Halsne et al reported that transfemoral amputees averaged only 1,540 prosthetic steps per day, indicating low levels of activity by transfemoral amputees [13].

The final two mobility limitations discussed here are higher energy consumption during gait (also described as decreased walking efficiency), and reduced walking speeds. It has been shown that energy consumption during gait is significantly higher in lower limb amputees than in non-amputees, and that the consumption is higher depending on the level of amputation (i.e., transfemoral amputees have a greater consumption than transtibial) and the cause (i.e., vascular amputees have a higher energy cost than traumatic amputees) [14]–[16]. In a study by Genin et al, the measured net energy consumption rate for transfemoral amputees was 30–60% greater than in non-amputee participants [16]. Similarly to energy consumption, walking speed is reduced in amputees as compared to non-amputees, and the speed reduces further for higher levels of amputation (i.e., transfemoral amputees) [16]. In their study, Genin et al also measured maximal sustainable speed, and found that the speed was approximately 1.2 m/s in transfemoral amputees, while greater than 2 m/s in non-amputees [16].

1.2.2 Overview of Prosthetic Knee Technologies

Among the improved modern treatments for amputation is the further development of prosthetic technologies, including prosthetic knees. These prostheses aim to improve overall mobility by providing better gait performance for transfemoral amputees, including greater efficiency and safety [5]. The following section will provide a review of prosthetic knee technologies.
1.2.2.1 Mechanical vs. Microprocessor/Electromechanical Knees

A variety of prosthetic knees exist for transfemoral amputees. Some prostheses have net power delivered to them (such as microprocessor-controlled or electromechanical knees) [17]–[19]. These prostheses have several advantages, including faster walking speeds, lower energy expenditure, decreased gait deviations and improved user satisfaction and quality of life [5], [20]. However, these more advanced, higher performing knees can cost up to five times that of a basic mechanical knee [21], which makes them less accessible to some patients, especially those in low income areas and developing countries. Consequently, most individuals rely on lower-level, mechanical knee joint technologies [5], [22]. Therefore, because mechanical knees are most commonly used by individuals with an amputation, this project focused on mechanical prosthetic knees only.

1.2.2.2 Stance Phase Control

Stance phase knee stability is achieved by resisting knee flexion during weight bearing, which is termed stance phase control [23]. Mechanical prosthetic knees are often prescribed for transfemoral amputees and these knees may incorporate a controller such as a load-dependent locking or braking function, which provides stability during stance by locking the knee in extension at initial contact [22].

1.2.2.3 Design of Prosthetic Knees and Effect on Mobility Limitations

For individuals with lower limb amputation, a prosthetic limb has the potential to compensate for the loss of mobility and permit them to reintegrate into their environment to pursue daily and social activity [12]. The following section will describe how the design of prosthetic knees can affect some of the mobility limitations that transfemoral amputees face.

One functional limitation in the design of many mechanical prostheses is the inherent trade-off between achieving weight-bearing stability and normal movements during gait [5]. This is because when a joint is designed to provide more stability (thus reducing falling and fear of falling), it inherently inhibits the smooth transition from stance to swing phase [5]. This can create a rigidity in gait, especially for lock- and brake-based controllers because of the knee movement restrictions. This can ultimately lead to an asymmetry in the gait, as the rigidly locked prosthetic limb becomes a pole over which the pelvis must vault [24]. An example of this is the
weight-activated braking knee, in which a brake remains activated into late-stance phase, which restricts flexion of the knee beyond what is normal for swing-phase initiation [24]. It has been shown that aligning or designing a prosthesis for maximum stability increases energy expenditure, possibly because the muscular effort required to initiate swing phase is increased to compensate for the strong locking/braking feature [5], [15].

In conclusion, while the design of the controller can provide stability in the knee, reducing the amputee’s rate of falling and fear of falling, the deviation from natural gait through delayed stance phase initiation or compensatory behaviour can increase energy expenditure and reduce walking speeds. To address these limitations, a mechanical knee was designed recently, called the All-Terrain knee, the features of which will be described below.

1.2.2.4 All-Terrain Knee

The All-Terrain Knee (AT-Knee) was developed in the Propel Lab and Bloorview Research Institute (see Figure 2). It is a mechanical knee prosthesis that makes use of an automatic stance phase lock (ASPL) controller. The ASPL controller provides a stabilizing function that does not interfere with swing-phase initiation [5]. This knee provides a high level of function, durability, ease of maintenance, and affordability for users [25]. It was chosen for this project because it aims to maximize some of the benefits of lock-based controllers, while minimizing the negative side effects described above.

The AT-Knee uses single-axis architecture with an automated mechanical lock to enable early-stance stability and late-stance flexion [25]. This monocentric mechanism operates by manipulating a knee lock to permit knee flexion and extension [26]. Similar to manually locking knees, the internal mechanism provides a completely locked knee joint in early and mid-stance phase [23]. Note, that while the knee joint itself is monocentric, it actually involves two axes, including the lock control axis and the knee axis [26]. The secondary control axis adds supplementary stability in stance phase by manipulating the knee lock, which was designed to engage and disengage to prevent or encourage flexion as needed based on the moment applied at the control axis [26]. The internal mechanism uses a knee lock that is automatically engaged when the knee extends and disengaged prior to swing-phase initiation; this provides stabilizing function of the knee that does not interfere with swing-phase initiation [5]. Disengagement of the
lock occurs as a result of an external knee extension moment, which normally occurs in mid to late stance phase [5].

Figure 3 shows how the knee lock articulates during different phases in the gait cycle to lock and unlock, permitting knee flexion and extension.

Figure 2: The All-Terrain Knee: (A) Picture of prosthesis assembly; (B) Labeled diagram of components; (C) Disengaged knee lock; and (D) Knee flexion of approximately 120 degrees [23].

Figure 3: Schematic of the AT-Knee during a typical gait cycle. The knee lock is engaged at the far left, preventing flexion in stance phase so that weight can be loaded. Then the knee lock begins to disengage as the gait cycle progresses to toe off, and the knee lock is completely
disengaged through swing phase. Finally, on the right the knee is locked again as the leg has swung forward and commenced a new gait cycle with heel strike.
Chapter 2

2 Background Literature

2.1 Gait Cycle

To understand gait analysis, it is important to understand gait, which is defined by the repetitive locomotion of the leg. A gait cycle is the period of time when one foot strikes the ground with the heel, swings through to advance forward through toe-off and make contact again with the heel. It is split into two phases: stance phase and swing phase. During stance phase, the leg moves from heel strike to toe off. Then in swing phase, the leg swings forward until the next heel strike. A schematic of a typical gait cycle can be seen in Figure 4.

![Gait cycle diagram](image)

**Figure 4:** A typical gait cycle [27]. Stance phase and swing phase are labeled for one gait cycle of the right leg. Stance phase occurs when the leg transitions through weight bearing of heel strike to toe off, and then swing phase occurs as the leg swings forward to make contact with the ground again at the subsequent heel strike.
2.2 Studying Prosthesis Function

2.2.1 Examples of Gait Analysis Techniques

Human gait can be analysed both qualitatively and quantitatively, but for research purposes, quantitative analysis is more beneficial when looking for trends and generalizing behaviour, as it is less subjective and reduces bias. More importantly, looking at specific parameters of interest can determine if the prosthesis is functioning as intended, and can help inform future design improvements. Currently, researchers tend to study biomechanics through gait analysis, and use various devices to measure specific parameters of interest [28]. These techniques and devices will be outlined briefly below.

One common method is video gait analysis, which uses cameras to record the gait of a walking patient [28]. Markers can be placed on points of interest on the body, or a markerless system can be used whereby software can be used to later determine regions of interest when analyzing the video. While markerless software reduces the need for a gait lab – because anyone can record the video using a standard file format and then process the video post-collection, thus allowing data collection in various environmental conditions – marked systems have significant limitations because they require stationary camera systems in a gait laboratory, limiting the environmental conditions that can be studied [29]. Both systems experience variability in results, inaccuracy and lack of reproducibility [29]. This is because the desired parameters, which is the kinematic data related to bony limb segments and joints, are calculated from the recorded data (surface tracking or recorded markers) and assumptions, not the data recorded [29].

Another method uses sensors positioned in the floor to measure pressures, forces and moments as patients walk on them [28]. The Ground Reaction Force (GRF) is the main force of interest that these systems measure and provides information about the center of pressure. The recorded pressure is given in percentage of weight and varies during the time the foot is in contact with the floor depending on where the patient is in stance phase [28]. These systems are not accessible to all researchers because they require a gait laboratory to be instrumented with floor sensors. Moreover, they require the study participant to walk in a specific location, which has the potential to alter the participant’s gait, misrepresenting their everyday activity pattern [29].
Lastly, more portable methods of analysis are being used whereby wearable sensors are placed on the body to measure the appropriate data [28]. For example, there are portable sensors that measure pressure and force (i.e., piezoelectric sensors and capacitive sensors), inertia (i.e., Inertial Measurement Units that use accelerometers or gyroscopes), and even current (i.e., Goniometers which change in resistance depending on flexion of the sensor) [28].

In many gait analysis studies, researchers utilize more than one of the above systems. One example is a study done by Bellmann et al. The group determined the performance of prosthetic knee joints using an optoelectronic camera system combined with two force plates as well as a mobile spiroergometric system. To identify the joint axes, passive markers were applied to each subject at anatomic reference points. The study was conducted in a gait laboratory under three environmental situations: walking on level ground, ramps and stairs [30]. While this system allowed for testing in various environments, it was still restricted to a gait laboratory in a clinical setting because of the large quantity of equipment required.

### 2.2.2 Computational Modeling

The section above described commonly used experimental techniques for understanding prosthetic function, and while those are frequently used methods, computational modeling can also be used. A computational model characterizes a system of study (i.e., the design of a prosthesis) using a variety of input parameters. It is a useful tool because the behavior of the prosthesis can be studied before putting time, effort and money into producing a physical prototype. These models often require empirical data, as was the case in [31] where Unal et al captured kinetic data from an amputee for their model. Inputting this empirical data can help facilitate design via the model, and optimize the design. For example, Shandiz et al created a two-dimensional model to investigate the dynamic characteristics of normal and transfemoral amputee locomotion for three types of motion controllers (frictional, elastic and hydraulic) and their design parameters were optimized to achieve the closest kinematics to that of the normal gait [32]. Similarly, Pejhan et al developed a dynamic model of a prosthetic knee controller to optimize the design to achieve the closest knee flexion pattern to that of normal gait [33].
2.3 Characterizing Stance Control Function of AT-Knee

Andrysek et al developed a new modeling method to characterize the stance control function of the AT-Knee [34]. The technique captured input variables using a wearable instrument which were then inputted into a model to translate these into the controller design [34].

2.3.1 Modeling Method

The function of the All-Terrain Knee ASPL controller is based on the moments and movements at the knee and control axes, the lock position and the knee position (extended/flexed). This set of four input variables can be reduced, due to the dependence between lock position and knee position as it can be assumed that the lock can only be engaged when the knee is fully extended [34]. Therefore, Andrysek et al defined the model based on KA and CA moments, as well as CA movements in terms of lock position [34]. The lock position was defined as $p$, and characterized by a displacement from fully unlocked ($> p0$) to a locked position between $p0$ and $p1$ [34]. $M_{CA}$ and $M_{KA}$ were defined as the moments about the control and knee axes, respectively [34]. $M_{CALc}$ and $M_{KALc}$ were the translated moments, as measured by the load cell (defined in Equations (3) and (4)) [34]. The model was also defined by the spring forces in the AT-Knee ($F1 = 35$ N and $F2 = 30$ N) [34]. Vertical and horizontal offsets from the KA and CA axes were defined as $d1 = 100$ mm, $d2 = 30$ mm, and $d3 = 16$ mm [34]. Figure 5 shows the placement of the input variables and constants used in the model. The model was defined by the relationships in the following equations:

$$M_{CA} = M_{CALc} - F_1 \times \Delta d_1 + F_2 \times \Delta d_2$$  \hspace{1cm} (1)

$$M_{KA} = M_{KALc} + F_2 \times \Delta d_3$$  \hspace{1cm} (2)
2.3.2 Overview of ASPL-SS

2.3.2.1 ASPL-SS Instrumentation

The Automatic Stance Phase Lock-Sensing System (ASPL-SS) was used to collect input variables for the model described above. The main componentry was a six-degree-of-freedom load transducer (ATI Mini58; ATI Industrial Automation, Inc., Apex, NC, USA) which provided direct measurements of external forces and moments acting on the limb [34]. The authors affixed the transducer below the AT-Knee using a standard pyramid connector with a custom adjustable adaptor for translational adjustments to the prosthesis alignment between the prosthetic knee and foot (see Figure 6.D) [34]. The moments at KA and CA were calculated based on the offsets distances between the transducer y-axis, the ASPL control axis ($\Delta x_1 = 23.5$ mm, $\Delta z_1 = 64.3$ mm), and knee axis ($\Delta x_2 = 16.5$ mm, $\Delta z_2 = 195.3$ mm), the torque about the y-axis ($TY$), the force along the x-axis ($FX$), the force along the z-axis ($FZ$), and using the following equations [34]:

\[
M_{CALc} = TY + FX \times \Delta z_1 + FZ \times \Delta x_1 \tag{3}
\]

\[
M_{KALc} = TY + FX \times \Delta z_2 + FZ \times \Delta x_2 \tag{4}
\]
Figure 6: Picture of ASPL-SS sensor placement and mounting. (A) Inductive proximity sensor, (B) force-sensing resistor, (C) accelerometer, and (D) load transducer [26].

Additionally, an inductive proximity sensor (DW-AD-509-509-M8-390; Contrinex, Givisiez, Switzerland) was positioned orthogonally to the steel knee lock component to continuously measure the anterior-posterior (i.e., locking and unlocking) movement of the lock throughout the gait cycle (see Figure 6.A) [34].

The APSL-SS had two additional sensors. First, a force-sensing resistor (FlexiForce A201; Tekscan, Inc., South Boston, MA, USA) was placed at the contact surface between the thigh component and the knee body (see Figure 6.B) to determine whether the knee was fully extended or flexed, where zero and non-zero forces corresponded to flexion and extension at KA, respectively [34]. However, the force-sensing resistor was not used in the determination of extension or flexion as the authors experienced inconsistent measurements [34]. Secondly, a linear accelerometer (ADXL 335; Analog Devices, Inc., Norwood, MA, USA) was affixed to the anterior surface of the All-Terrain Knee (see Figure 6.C) to record anterior, posterior, and vertical components of acceleration at the proximal end of the prosthetic shank. The primary use of the accelerometer was to synchronize the accelerometer, proximity sensor and force-sensing resistor with the load transducer [34].

For data collection purposes, the load transducer data was transmitted and viewed wirelessly using Java software [34]. The proximity sensor, force-sensing resistor and accelerometer were connected using an Arduino UNO, which saved the sensor outputs locally on a MicroSD card,
and also transmitted wirelessly to a nearby laptop via a wireless radio frequency (RF) module (XBee Pro Series 1; Digi International, Minnetonka, MN, USA) [34]. The system was powered by a six AA and two 9V batteries [34].

2.3.2.2 Data Collection Using ASPL-SS

The ASPL-SS was used to collect data for able-bodied and amputee participants. Testing involved eight able-bodied participants, and one highly-active unilateral transfemoral amputee [34]. The able-bodied participants wore a prosthetic simulator to mimic the function of the prosthesis [34]. This simulator was adjusted to fit each participant by the research team who was trained by a prosthetist, and the participants completed warm-up walking trials prior to collecting data [34].

2.3.3 Data Analysis

The binary states (positive/negative) of the three input variables were used to calculate the AT-Knee controller state (see Table 1 for the 8 states) [34]. Then, each step was isolated and normalized to 100% of the stance phase and occurrences of states determined as percentages of stance phase [34].

<table>
<thead>
<tr>
<th>Control State</th>
<th>KA External Moment State</th>
<th>CA External Moment State</th>
<th>Lock State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ext</td>
<td>Flex</td>
<td>Ext</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 1: Possible states of the three input variables, and the resulting control state [34].

2.3.4 Results of Characterizing Stance Control

The expected controller states of the system were states 2, 3, 6 and 7 [34]. These translated to stages of stance phase, as seen in Figure 7. Andrysek et al found that the ASPL controller was predominantly characterized by the following control states: state 2 (knee extension moment), state 6 (controller is free to flex; pre-swing), and state 7 (knee locked and providing stability).
For the simulated gait, the expected states defined between 92.9% and 94.7% (mean 93.6 and SD 2.2) of the stance phase model [34]. Inclusion of states 4 and 8 increased mean model accuracy to up to 99.7 (SD 0.4) % for simulated gait [34].

**Figure 7:** Expected states of the 8-state model, and the corresponding stage in stance phase. Adapted from [34].

In the application of their model, Andrysek et al also tested the robustness, responsiveness and predictive capability of their model [34]. For example, the authors introduced variance in the moment and lock position measurements to assess the robustness of their model, and recalculated the percentage of stance phase accounted for by the expected states and combined states [34]. Ultimately, the authors provide a novel technique to characterize, predict and optimize the design of a prosthetic controller.
Chapter 3

3 Statement of Research Problem and Objectives

3.1 Research Problem

Section 2.2 outlined examples of commonly used gait analysis techniques. While kinetic and kinematic gait data is commonly collected to measure the performance of prostheses for amputees, the collected data does not provide direct feedback about whether the controller mechanism is performing as intended. This is because these techniques use external systems to infer information about function of the prosthetic knee rather than directly monitoring the internal mechanisms. Therefore, there is also merit to understanding the inner workings of a prosthetic knee as information about the controller can be used to inform future design decisions. Since many amputees are prescribed a passive knee [22], characterizing the function of these knees is valuable for a large portion of the patient population. In characterizing the function of the knee, the design of the prosthetic knee can be optimized.

Another major limitation of many of these techniques is that they restrict the testing environment. Instability of gait more commonly affects amputees in their day-to-day lives as they walk over a variety of terrains in the community, and data collected on level ground in a clinic or gait laboratory is not indicative of the instability they experience while walking on outdoor uneven ground.

Previous effort by Andrysek et al (see Section 2.3) showed the feasibility of using portable sensors to collect empirical data and input that into a model to characterize function of the AT-Knee. However, the mentioned system required the disassembly of the prosthetic leg to insert a load transducer for data collection, which makes the setup of the data collection system more cumbersome. Moreover, there was no single unit that contained the electronics of the system, also increasing the workload of setting up the data collection system, as each sensor needed to be mounted individually.

3.2 Goal

The goal of this project was to improve on the empirical data collection system (known as the wearable gait analysis tool) designed by Andrysek et al [26], [34]. The purpose of the wearable
gait analysis tool is to capture data about the function of the knee, and to use that data to characterize the knee function. The final goal of the completed system is to collect empirical data about the AT-Knee function, which can be fed back into a computational model to make design improvements to the AT-Knee.

3.3 Objectives

The primary objectives of this work were as follows:

A. Perform a literature review to determine the current methodological approaches to evaluating and designing prosthetic knees;

B. Design a self-contained, wearable gait analysis tool (see Table 2 for specific design requirements);

C. Validate, through benchtop engineering validation tests, that the inductive proximity sensor can be replaced by a linear position sensor;
   a. Test 1: Verify that a linear position sensor accurately measures linear movement experimentally, as compared to manufacturer’s expected output.
   b. Test 2: Verify that a linear position sensor measures linear movement under hysteresis testing (repeated small movements).
   c. Test 3: Verify that a linear position sensor accurately measures lock position, as compared to the inductive proximity sensor (IPS) used by Andrysek et al [26], [34].

D. Measure accuracy of the wearable gait analysis tool sensors, and
   a. Test 1: Compare the accuracy of the pressure sensor and angled linear position sensor to the knee axis moment measured by the load cell in detecting extension and flexion during stance phase, to determine which of the two sensors can replace the load cell in detecting extension and flexion about the knee axis.
   b. Test 2: Compare the accuracy of the linear position sensor to the control axis moment measured by the load cell in detecting extension and flexion during
stance phase, to determine if the linear position sensor can replace the load cell in detecting extension and flexion about the control axis.

E. Develop 4 finite state outputs using sensors of the wearable gait analysis tool

a. Calculate the mean percentage of stance phase for the 4 states, and the states of the 8-state model.

b. Determine which states of the 8-state model are best represented by the 4 finite states.

i. Test 1: Perform paired t-tests to detect any significant differences.

ii. Test 2: Take the mean difference between the 8-state model and the 4 finite states to find the smallest mean difference.

The secondary objective of this work was to increase usability of the wearable gait analysis tool by designing a simple user interface that guides the user through the process of collecting gait data. The secondary objective was completed in BME1802, Human Factors of Medical Devices. More information about this can be seen in Appendix D.

3.4 Design Requirements

Below is a table of the design requirements that must be fulfilled during technical development of the wearable gait analysis tool.

Table 2: Design requirements for the technical development of the wearable gait analysis tool

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight as minimal as possible</td>
<td>$\leq 400$ g$^1$</td>
</tr>
<tr>
<td>Dimensions must be less than or equal to the AT-Knee</td>
<td>Width, height</td>
</tr>
<tr>
<td>Self-contained unit attaches to and detaches from the AT-Knee without requiring modification to the setup of the prosthesis components</td>
<td></td>
</tr>
<tr>
<td>Attachment of tool to AT-Knee should take no more than one person</td>
<td>Number of people required to setup tool</td>
</tr>
<tr>
<td>Sensors should be placed in the same position every time the tool is used</td>
<td>Location of sensors</td>
</tr>
</tbody>
</table>
Weight must be below 400 g to limit the effect on the amputee’s gait while wearing the wearable gait analysis tool. Reducing added weight to the prosthesis will limit the effect the wearable gait analysis tool has on altering the biomechanics of the amputee’s gait [35].
Chapter 4

4 Literature Review

4.1 Methodology

To formulate the research problem, a literature review was conducted on the design of energetically-passive, mechanical prosthetic knees, specifically focusing on the evaluation of the performance of the prostheses.

The following databases were searched: PubMed, IEEE Xplore, Scopus, and Web of Science. Searches contained a variation of the following terms, depending on the database: ("prosth* knee" OR "knee prosth*" ) AND amput* AND design ) AND transfemoral NOT active NOT power*. Searches were limited to human studies. Moreover, additional sources were handpicked as grey literature, and additional papers were selected from the references of the original papers. Searches were performed covering the period of the earliest date through March 2018.

Records were included if abstracts met the following conditions: discussed the design of mechanically passive knee prostheses; evaluated gait only; and involved transfemoral amputees only. Records were excluded if they discussed microprocessor or powered knees, evaluation methods on squatting or sitting, or focused on clinical testing. More specifically, all comparative and random clinical studies were excluded as clinical measures do not directly inform design nor is that usually the purpose of those papers. Only English texts were reviewed. This screening was performed by reviewing title and abstracts, and where the abstract did not provide sufficient information to make a decision, the full text was reviewed. Full-text articles were reviewed by one reviewer based on the same criteria for the title and abstract review to further refine the search results.

Data were synthesized by reviewing the full text of all remaining publications and categorizing the studies in terms of their stage in the design cycle. Papers were classified into the four stages of the design cycle, but the work focused on the device design and validation as this is more pertinent to informing engineering design and specifically the research question of this thesis. The Creating Solutions stage was further delineated into: (i) Computational solutions and (ii) Physical prototyping. Similarly, the Preliminary Evaluation stage was broken into four categories: (i) Static computational evaluation, (ii) Dynamic computational evaluation, (iii)
Benchtop experiments and (iv) Clinical experiments. Clinical experiments that included qualitative measures (i.e., surveys or interviews) were excluded. Moreover, any publications that did not discuss at least one stage of the design cycle were excluded.

**Figure 8:** Iterative design cycle, modified from [36]–[38].

### 4.2 Results

The papers included in the review were classified based on the stage of the design cycle (Figure 8) that they discussed. These results are found in Table 3.

**Table 3:** Results of literature review. Each paper is classified by the phases of the design cycle discussed in the paper, indicated by a checkmark. The papers are sorted chronologically.

<table>
<thead>
<tr>
<th>Author</th>
<th>Year</th>
<th>Analyzing Needs</th>
<th>Developing Ideas</th>
<th>Creating the Solution</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Computational</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Statics</td>
</tr>
<tr>
<td>Wallach [39]</td>
<td>1970</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hobson [40]</td>
<td>1974</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Cortesi [41]</td>
<td>1975</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hobson [42]</td>
<td>1975</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zarrugh [43]</td>
<td>1976</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cappozzo [44]</td>
<td>1980</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fisher [45]</td>
<td>1985</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Koganezawa [46]</td>
<td>1987</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diangelo [47]</td>
<td>1989</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Farber [48]</td>
<td>1995</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blumentritt [49]</td>
<td>1997</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blumentritt [50]</td>
<td>1998</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jin [51]</td>
<td>2003</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Andrysek [52]</td>
<td>2005</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Andrysek [53]</td>
<td>2007</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pejhan [33]</td>
<td>2008</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unal (1) [54]</td>
<td>2010</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unal (2) [55]</td>
<td>2010</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Andrysek [23]</td>
<td>2011</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Behrens [56]</td>
<td>2011</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authors</td>
<td>Year</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>--------------------</td>
<td>------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Arnout [57]</td>
<td>2012</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unal [58]</td>
<td>2012</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hamner [59]</td>
<td>2013</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inoue [60]</td>
<td>2013</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Unal [31]</td>
<td>2013</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Arelekatti (2) [25]</td>
<td>2015</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Arelekatti (1) [61]</td>
<td>2015</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Arelekatti [63]</td>
<td>2016</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Bhuiyan [64]</td>
<td>2017</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Phanphet [65]</td>
<td>2017</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Computational modeling as a strategy for creating or evaluating the solution was discussed in 18 of the 31 reviewed papers. Computational modeling is a design strategy that can help the design meet certain specifications by evaluating the stability features of different knee mechanisms or assessing the performance of possible controllers without actually constructing a prototype [43]. It is advantageous because changes can be made to the model to assess outcome on the design without significant added cost or time. Moreover, instead of finding the mechanical properties of the controllers by experimental tests or through numerical trial and error, an optimization method can be followed to obtain the optimal values that result in a kinematic pattern close to the normal gait [33]. Computational models can also be used as a basis for the evaluation of an existing design, as was reported in the literature [52], [64], [65]. For example, Andrysek et al used knee
instability diagrams to evaluate the stability of their prototype prosthesis compared to other existing knee prostheses [52]. Using load line vectors to define zones of instability, Andrysek et al use the union of two zones of instability to determine the overall zone of instability, which depicts the stability characteristics of each prosthetic knee [52]. When a model is developed, it is good practice to validate the model. To validate the computational model, several papers compared the model with known experimental results or healthy human gait [39], [43], [55], [58]. In the case of Wallach et al, the walking pattern for a healthy man was obtained from the literature and used to compare to the force, velocity and power as calculated for the control mechanism by the model [39]. Some papers used information gathered in experimental evaluation to inform their computational model and ultimately evaluate performance of their prosthesis [31], [47], [64]. Bhuiyan et al recorded the gait cycle of a healthy individual to obtain spatial, temporal, kinematics, and kinetics information of lower limb locomotion [64]. This data was also used as a comparison between the experimental and predicted data, to determine the performance of the prosthesis in the model [64].

Experimental evaluation of a physical prototype was also seen in literature. During experimental evaluation, a physical prototype was either used in benchtop testing with equipment to test physical characteristics of the prototype [45], or in clinical testing with participants actually using the prototype under specified conditions [23], [25], [53], [54], [60], [63], [64], [66], [31], [45]–[51]. In benchtop tests described by Fisher et al, to examine the integrity of their prosthetic knee, units were subjected in a machine to repeated cycles of loading, chosen to represent the conditions of normal level walking [45]. Clinical evaluation was found to be a more common method of evaluation. Two common technologies used in clinical evaluations were force plates [31], [45], [47], [50], [53], [54], [60] and video analysis tools [31], [47], [51], [53], [54], [60], [64].

4.3 Discussion

Once the design is in the prototyping stage, evaluation of the prototype becomes essential. Although clinical experimentation proves to be a widely used strategy for evaluating prototypes, it also has limitations. A large number of papers used biometric data, such as physiological cost index [5], [23] or forces and torques of associated musculature [54], as a proxy for performance of the prosthesis in their experiments. While this indicates clinical performance of the prosthesis,
it does not give direct feedback to the designing engineers about how their design is performing or what specific aspects of the design need to be changed to improve performance. The lack of feedback about how the design is performing according to many of the original design requirements is troublesome because it creates a disconnect between the information provided during validation and the interpretation by the engineer. One proposed solution might be to complete more benchtop testing as this is a more common method to test some of the design requirements. However, this does not provide information about the performance when an amputee wears the prosthesis. This is equally troublesome because different users adopt different walking patterns, and benchtop testing does not allow for testing of the prosthesis while in use under a variety of different terrain conditions (i.e., stairs, grass, rocks). Breakey et al acknowledged that they were unaware of any studies that guided selection of the most appropriate parameter(s) to improve the prosthetic control by the amputee [67]. In fact, practitioners mainly relied on clinical experience and manufacturer’s recommendations to attempt to obtain the most efficient and balanced gait [67]. Studies on the performance of prosthetic knees could fill this need to determine which parameters are appropriate for specific patients.

Another common limitation in the literature review was the low number of amputee participants, with the average number of participants being approximately 2.4. Consequently, this is another significant limitation of the design cycle, as results from clinical evaluation that feed back into the design of the prosthesis will be largely based on a small sample size, not indicative of the larger population. Thus, it is recommended to increase sample sizes of amputee participants to capture variation in walking patterns while using prosthetic knees.

As seen in Table 3, to create a design solution, several papers discussed using computational modeling, and also used that model to validate the design. Contrarily, a majority of the papers described experimental methods (either benchtop or clinical) that evaluated the design. However, few papers used both methods. Furthermore, a possible effective way to validate the design of the knee would be to collect data from a physical prototype and input the data into the design model for optimization of the knee.
Chapter 5

5 Design and Benchtop Testing

5.1 Methodology and Design

5.1.1 Design of WGAT Instrumentation

A wearable gait analysis tool was designed to be standalone and provide temporary attachment to the All-Terrain Knee with the purpose of supporting gait analysis in a wide variety of environments.

5.1.1.1 Measuring Knee Lock Position

A linear position sensor (LPS) was chosen to measure the anterior-posterior position of the lock. The LPS chosen was the 9605 Miniature Spring Return Linear Motion Position Sensor by BEI Sensors. This sensor was mounted perpendicular to the knee lock such that it had contact with the knee lock at all times during gait, and the movement of the sensor plunger was within the range of electrical travel as specified by the manufacturer (see Appendix E).

This sensor was chosen to replace the inductive proximity sensor (IPS) because of lower weight and size of the sensor, as well as reduced power requirements that required a slighter and smaller battery.

Figure 9: Placement of the LPS [68], perpendicular to the knee lock. Image of the LPS is enlarged for visibility.
5.1.1.2 Knee Flexion/Extension Detection

Two sensors were used to detect knee flexion and extension. The first was a FlexiForce A201 force-sensing resistor (response time: \( \leq 5\mu s \); 0-110N; thickness: 0.208mm; length: 51mm; sensing \( \Omega \): 9.53mm) (Tekscan, Inc.; South Boston, Massachusetts) that was placed between the contact surfaces of the thigh and knee body of the AT-Knee. Because of the thin profile of the force-sensing resistor, the gap between the knee surfaces and the sensor resulted in binary signals for hyperextension and flexion [26], [34]. To minimize the gap, a rubber buffer was affixed to the force-sensing resistor to obtain signal readings for flexion and extension.

The second sensor was an additional LPS that was placed at an angle to the thigh portion of the knee.

5.1.1.3 Accelerometer

A linear accelerometer was also incorporated into the system (0.5-550Hz; \( \pm 3g \); sensitivity: 360mV/g) (ADXL 335 Analog Devices, Inc.; Norwood, Massachusetts) to record acceleration in the x-, y-, and z-directions. The accelerometer was used for data synchronization between the WGAT and load transducer, by detecting stomp data.

5.1.1.4 Data Logging and Transmission

Measurements from the linear position sensors, force-sensing resistor and accelerometer were collected by an Arduino UNO. An Arduino Wireless/SD Shield was mounted onto the Arduino UNO, and was used to collect sensor readings and store them in a comma separated values (.csv) file on a microSD card.
5.1.1.5 Power Supply

Power for the electronics in the WGAT was provided by a rechargeable LiPo battery cell (3.7V, 1200mAh).

5.1.1.6 Sensor Mount and Electronics Containment

Two prototypes were created for the sensor mount and electronics containment.

5.1.1.6.1 Prototype 1

The concept for the first prototype involved a single housing, which contained the sensors and all associated electronics. The housing was 3D-printed in white PLA plastic. A dremel was used to create holes in the cover of the AT-Knee for the sensors to pass through, and then the housing was affixed to the cover with epoxy. The housing could then be attached and detached from the
AT-Knee using pre-existing mechanisms, which have been shown to require only a single person to attach to the AT-Knee. Figure 11 shows images of Prototype 1.

In creating the first prototype, benefits and shortcomings were discovered, which informed the design decisions for Prototype 2. These are summarized in Table 4.

**Table 4**: Summary of benefits and shortcomings of Prototype 1.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Benefits</th>
<th>Shortcomings</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D-printed PLA</td>
<td>Light-weight</td>
<td>Less accuracy for small features</td>
</tr>
<tr>
<td></td>
<td>Rapid prototyping</td>
<td></td>
</tr>
<tr>
<td>Housing interfaces with existing attachment mechanisms</td>
<td>Uses pre-existing attachment mechanisms that are known to be secure</td>
<td></td>
</tr>
<tr>
<td>Attached with epoxy</td>
<td>Rapid prototyping</td>
<td>Weak adherence between the two plastics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Less secure than other attachment mechanisms (i.e. screws)</td>
</tr>
<tr>
<td>Single container with all electronics</td>
<td>Wires less prone to breakage because they do not move once inside the housing</td>
<td>Increased weight at the knee</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Electronics are less accessible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Housing is large and bulky</td>
</tr>
<tr>
<td>Open cavity in the housing</td>
<td></td>
<td>Sensors are less secure because there are many degrees of freedom for them to move</td>
</tr>
</tbody>
</table>
Figure 11: First prototype of the sensor mount and electronics containment. All electronics and sensors were housed in a single, 3D-printed container. (left) Front view. (right) Side view.

5.1.1.6.2 Prototype 2

Learning from the shortcomings of Prototype 2, a second prototype was designed. An engineer from LegWorks, Calvin Ngan, modeled the prototype in SolidWorks, and the author 3D-printed the prototype in PLA plastic.
Table 5: Overview of features in Prototype 2 that address the shortcomings of Prototype 1.

<table>
<thead>
<tr>
<th>Shortcomings</th>
<th>Solution in Prototype 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attached with epoxy:</td>
<td>Attached with 3 screws</td>
</tr>
<tr>
<td>• Weak adherence between the two plastics</td>
<td></td>
</tr>
<tr>
<td>• Less secure than other attachment mechanisms</td>
<td></td>
</tr>
<tr>
<td>Single housing:</td>
<td>Two-piece housing created. Off-the-shelf box attached to thigh to hold associated electronics, and a 3D-printed mount that allows sensors to be attached on the outside making them more accessible.</td>
</tr>
<tr>
<td>• Increased weight at the knee</td>
<td></td>
</tr>
<tr>
<td>• Electronics are less accessible</td>
<td></td>
</tr>
<tr>
<td>• Housing is large and bulky</td>
<td></td>
</tr>
<tr>
<td>Sensors are less secure because there are many degrees of freedom for them to move</td>
<td>Pockets created for sensors to secure them on multiple sides, and then sensors were secured with additional screws.</td>
</tr>
</tbody>
</table>
**Figure 12:** Schematic of the second prototype of the sensor mount. *(A) Front view.* Arrow indicates the where the accelerometer attaches to the mount via screws. *(B) Side view.* Top arrow indicates the channel to hold a single cord with the wires that leads to the box on the thigh. The bottom arrow indicates the pocket for the LPS that measures lock position. *(C) Side back view.* The top arrow shows the curved design of the mount that fits with the AT-Knee cover, and the attachment mechanism via screws. The bottom arrow indicates the pocket for the angled LPS.

The second prototype had an off-the-shelf box that attached to the thigh with two Velcro straps. The box contained all the associated electronics, including the circuit board, Arduino Uno and SD shield, and lithium battery. A single cord connected to a 3D-printed mount, and was secured in a channel on the mount. This contained wires that branched out to connect all the associated sensors. The 3D-printed mount attached to the AT-Knee cover with 3 screws. It contained two pockets that secured the LPSs from the sides and back, and then a screw was drilled into the cover to secure the LPSs from the front. The pocket for the angled LPS was designed such that the LPS captured approximately the last 10 degrees of movement in the articulating portion of the AT-Knee, but also so that the plunger of the LPS made contact without being crushed when the knee was extended. The accelerometer was attached to the front of the 3D-printed mount.
with four screws. Finally, the force-sensing resistor was inserted through a slot on the 3D-mount and secured with tape. The mount was designed to fit the sensors such that they were placed on the AT-Knee in the same position every time.

Figure 13: Setup of second prototype on an able-bodied participant. The second prototype included a 3D-printed mount for the sensors, and an off-the-shelf plastic box for the associated electronics, which was secured to the thigh with Velcro straps.

The width and height of the 3D-printed sensor mount were 62.12 mm and 93.63 mm as measured by a Mitutoyo caliper. Conversely, the dimensions of the AT-Knee are 77.80 mm and 186 mm. The weight of the entire wearable gait analysis tool was measured to be 352 g, with the section mounted on the knee weighing only 140 g.

5.1.2 Engineering Validation Tests

5.1.2.1 Calibration of Linear Position Sensor

To determine the performance of the linear position sensor, the experimental readings of the linear position sensor were compared to the calculated theoretical values. The theoretical values
for the sensor were calculated based on the output ratio in the technical documentation (see Appendix E), and a theoretical equation of:

\[ y = 80.55x + 4 \times 10^{-13} \]  

(5)

Where \( y \) is the sensor output in bits, and \( x \) is the amount the plunger has been depressed in mm.

The protocol for experimental data collection was adapted from tests previously done with the inductive proximity sensor [69].

5.1.2.1.1 Experimental Design

A flat block was secured in a vise on a milling machine, which moved in precise increments of 0.01 mm, as displayed on the mill’s digital readout. The linear position sensor was secured in the chuck of the mill, perpendicular to the flat block, and remained stationary as the flat block was moved in relation to the sensor. Two tests were performed with the sensor: a distance test and a hysteresis test.

![Side View](image)

**Figure 14:** Setup of the flat block in the vise, with the sensor secured in the chuck of the milling machine [69].

5.1.2.1.2 Distance Test

During the distance test, the vise was moved in 0.02 mm increments from 1.24 mm to 5.24 mm, for a total of 4 mm of travel. This distance was chosen because it was within the electrical travel distance for the sensor (see Appendix E). The output bits at each 0.02 mm increment were
manually recorded. This was repeated three times, and the average of the three trials was calculated.

5.1.2.1.3 Hysteresis Test

A hysteresis test was performed to determine if the repetitive anterior-posterior movement of the knee lock would have an effect on the sensor [69]. The vice was moved back and forth from 2.35 mm to 2.85 mm in 0.10 mm intervals. Similarly to the distance test, this was repeated for three trials.

5.1.2.2 Dynamic Testing with LPS and IPS

Dynamic testing was completed by mounting the LPS and IPS on a knee assembly at the same time, so that the lock movement measured by the two sensors could be compared directly. The purpose of this test was to determine if the LPS provided similar or the same measurements as the previously used IPS. The knee assembly was manipulated to simulate the following conditions: normal walking, normal gait with heavy heel strike, running and tiptoeing. Heavy heel strike was characterized by a stronger impact between the heel and the ground upon heel strike, running was characterized by quicker manipulation of the knee assembly to simulate a more rapid gait cycle, and tiptoeing was characterized by a softer contact upon heel strike. Each condition was repeated for multiple simulated gait cycles. The outputs of the LPS and IPS were recorded on an Arduino Uno.

5.2 Results of Engineering Validation Tests

5.2.1 Calibration of Linear Position Sensor

Figure 15 and Table 21 in Appendix F show the results from the distance test for the linear position sensor. As compared to the theoretical values with a ±3% error, the average experimental results were within the range of the expected values.
Figure 15: Results from the calibration of the linear position sensor for the distance test.

Figure 16 shows the individual results from the hysteresis test for each of the three trials with the linear position sensor. Similar to the distance test, the theoretical values for the sensor have error bars for ± 3% expected error. Each reading was within the manufacturer’s specification for the sensor.

Figure 16: Results from the calibration of the linear position sensor for the hysteresis test.

The results from the distance test and the hysteresis test showed that the linear position sensor performed within the manufacturer’s specifications, producing sensor outputs that match the expected values.
5.2.2 Dynamic Testing with LPS and IPS

Figure 17, Figure 18, Figure 19, and Figure 20 show the results for the dynamic testing of the LPS and IPS. The conditions shown in the figures are simulated normal walking, simulated normal walking with heavy heel strike, simulated running and simulated tiptoeing, respectively. The root mean square error were 0.00968, 0.02917, 0.02002, and 0.01294, respectively. The root mean square error between the LPS and IPS was lowest for simulated normal walking, and highest for simulated normal walking with heavy heel strike.

![Figure 17](image1.png)

**Figure 17:** Lock position as measured by the linear position sensor and inductive proximity sensor for simulated normal walking.

![Figure 18](image2.png)

**Figure 18:** Lock position as measured by the linear position sensor and inductive proximity sensor for simulated normal walking with heavy heel strike.
5.3 Discussion

5.3.1 Calibration of LPS

In the preliminary validation tests with the linear position sensor, the experimental outputs from the LPS performed within the calibration curves provided by the manufacturer. Results of the calibration tests showed that the sensor accurately measured distance changes, both in sequential linear distance tests, and in the hysteresis tests. The results from the hysteresis tests were important in showing that the LPS would accurately measure distance when repeatedly moved back and forth within a small range, as would be the case when used with the repetitive gait pattern of a user walking with the AT-Knee.
5.3.2 Dynamic Testing between LPS and IPS

The root mean square error between LPS and IPS was highest for the simulated normal walking with heavy heel. This might be because one of the major differences between the LPS and IPS is that the LPS makes contact with the knee lock, whereas the IPS determines lock position without making contact with the lock. In heavy heel strike, the knee assembly was articulated with higher impact upon heel strike, and the movement of the lock and knee assembly might have moved the LPS during data collection.
Chapter 6

Measuring WGAT Function and 4 Finite States

6.1 Methodology

To measure WGAT function, walking trials were completed with able-bodied participants, who walked with a gait simulator instrumented with the WGAT and a load cell. Further, 4 finite output states were developed.

6.1.1 Portable Force and Torque Transducer

To validate the function of the wearable gait analysis tool, the prosthetic legs used for data collection were instrumented with an ATI Mini58 three-axis (six-degree-of-freedom) force and torque transducer (ATI Industrial Automation, Inc.; Apex, North Carolina) (Figure 21) [26]. Similar to the work by Andrysek et al [26], [34], a load transducer was chosen for validation because load transducers have been shown to correctly identify gait cycle events [70], [71]. The load transducer used by Andrysek et al was heavy and cumbersome as it required insertion between the knee and lower pylon. Therefore, comparing the detection of gait events from the sensors on the WGAT to the identification by load transducer will show the differences in accuracy between the WGAT, and the previously used system by Andrysek et al.

However, the load transducer has a different coordinate system origin than the All-Terrain Knee axes (knee and control axes, as seen in Figure 21 and Figure 22); therefore, the equivalent forces and moments acting on the control and knee axes were calculated by transforming the raw data (more detail can be found in the MATLAB code in Appendix B) [26]. Figure 22 shows a schematic that illustrates the offsets between the ATI Mini58 F/T Transducer Y-axis (red), the ASPL control axis (pink) [Δx1 = 23.5mm, Δz1 = 64.3mm], and the ASPL knee axis (maroon) [Δx2 = 16.5mm, Δz2 = 195.3mm] used to derive the moment applied at the control and knee axes from the forces and torques acting at the ATI origin (coordinate axes shown) based on the following equations:

\[ \text{Control Axis Moment} = TY + FX \Delta z1 + FZ \Delta x1 \]  \hspace{1cm} (6)

\[ \text{Knee Axis Moment} = TY + FX \Delta z2 + FZ \Delta x2 \]  \hspace{1cm} (7)
Where $TY$ represents torque about the y-axis (in the sagittal plane), and $FX$ and $FY$ represent force along the x- and y-axes, respectively [26].

Data were sampled at over 100 Hz using the transducer, transferred via a wearable battery-powered ATI Wireless F/T Transmitter (ATI Industrial Automation, Inc.; Apex, North Carolina) to a nearby laptop, and displayed by the ATI Wireless F/T Java software from which they were recorded to .csv files for analysis [26].

![Figure 21](image)

**Figure 21:** All-Terrain Knee instrumented with ATI Mini58 F/T Transducer using custom adaptor plates above and below the transducer. *(left)* Front view. *(right)* Side view [26].
**Figure 22:** Schematic illustrating the offsets between the ATI Mini58 F/T Transducer Y-axis (red), the ASPL control axis (pink), and the ASPL knee axis (maroon) [26].

Table 6 provides a summary of all the sensors used in this project, including the sensors in the WGAT and the sensor used for validation of the final system.

**Table 6:** Summary of sensors, what they measure, the purpose of the sensors and the conventions of the measurements.

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Measurement</th>
<th>Purpose</th>
<th>Conventions[^1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear position sensor</td>
<td>Anterior/posterior lock movement</td>
<td>Determine lock position</td>
<td>Unlocked &gt; threshold</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Locked &lt; threshold</td>
</tr>
<tr>
<td>Angled linear position sensor</td>
<td>Movement of thigh portion of AT-Knee</td>
<td>Determine moments of extension/flexion</td>
<td>Extension &gt; threshold</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flexion &lt; threshold</td>
</tr>
<tr>
<td>Accelerometer</td>
<td>$a_x, a_y, a_z$</td>
<td>Synchronization of WGAT and transducer</td>
<td>$\Delta a$ vertical &gt; 40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sync point</td>
</tr>
<tr>
<td>Force-sensing resistor</td>
<td>Force at contact point of thigh and shank portion of AT-Knee</td>
<td>Determine moments of extension/flexion</td>
<td>Extension &gt; threshold</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flexion &lt; threshold</td>
</tr>
<tr>
<td>Force/torque transducer</td>
<td>$F_Z, F_Y, F_Z, T_Y, T_Z, CA$ moment, $KA$ moment</td>
<td>Validation of the WGAT</td>
<td>$\Delta F_Z &lt; -30000$ sync point</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$KA/CA$ moment &gt; 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$KA/CA$ moment &lt; 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>extension</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>flexion</td>
</tr>
</tbody>
</table>
Thresholds were selected based on trial and error to maximize overall accuracy. More information is in Section 6.2.

6.1.2 Study Design

6.1.2.1 Participants

Able-bodied participants were chosen for this study because the applicable above-knee amputee population is limited; there was a pool of four All-Terrain Knee users from which could be drawn [26]. Moreover, this study focused on the engineering validation of the WGAT by determining how well the sensors of the WGAT performed compared to a load transducer, and thus clinical results were not of interest. In order to select a larger sample size, a convenience sample of seven able-bodied participants were recruited to test the WGAT while wearing a prosthetic gait simulator (Figure 23).

6.1.2.1.1 Inclusion Criteria

To be considered for participation in this experiment, participants had to:

1. be above the age of 18;

2. be able to communicate in English;

3. weigh less than the maximum loading capacity of any component of the instrumented prosthetic simulator (maximum 100kg/220lb);

4. be tall enough (at least 140cm/4’7”) to be fitted with a prosthetic simulator including an adult sized All-Terrain Knee, load transducer, and the applicable adapters; and

5. be strong, independent ambulators [26].
6.1.2.1.2 Recruitment

A convenience sample of able-bodied participants was recruited primarily from the University of Toronto. Potential participants were identified through self-referral, and once potential participants were identified, they were contacted, the experiment was described in more detail. Data collection sessions were scheduled at Holland Bloorview Kids Rehabilitation Hospital at convenient times for participants. Before commencing the sessions, written consent was obtained from each participant.

6.1.2.1.3 Ethical Considerations

Prospective participants were assured they were under no obligation to participate, and could withdraw their participation at any point during the study [26]. Moreover, they were informed of any potential risks associated with the study protocol, including possible discomfort caused by the simulator or initial instability when walking with the gait simulator [26]. Breaks were requested and taken as necessary by all participants [26].

Each participant was assigned a random identification (ID) code. The identifying information linked to the code was stored separately from the collected gait data in a locked cabinet. Only the principal investigator (Jan Andrysek) and the research coordinator (Rachel Reding) had access to the data. Upon completion of the study, the link between participant and ID code was destroyed.
Data were analyzed and presented using only the identification code. All personal information was kept confidential. If the results of the study are published, participant names and identifying information will not be used. Following the completion of the study, data will be saved in its anonymized state for seven years as required by Holland Bloorview, after which it will be destroyed.

This experiment was approved by the Holland Bloorview Research Ethics Board and University of Toronto Office of Research Ethics. Express written consent was obtained from all participants prior to beginning data collection.

6.1.2.1.4 Study Participants

Seven able-bodied participants (3 males and 4 females; age [mean ± standard deviation]: 24 ± 3 yr; weight: 66 ± 14 kg (measured without prosthetic simulator); height: 168 ± 10 cm) were recruited for participation in the study. One participant was excluded due to a disconnection of the data collection system during walking trials, resulting in a sample size of six. In relevant previous studies evaluating the use of load cells and prosthetic gait simulators, sample sizes ranged from one to 10 participants [70], [72], [73]. Moreover, the average number of participants in engineering validation studies from the literature review was 2.4.

6.1.2.2 Experimental Procedure

Able-bodied participants were fit with a gait simulator by the author. An adjustable pylon was used between the load transducer and the foot, which was adjusted to the appropriate height for the participant. Further, with the help of a certified prosthetist, two inserts were designed to reduce the diameter of the gait simulator socket. None, one or both of the inserts were used to adjust the size, according to the participant’s thigh size.

Before data collection, participants were provided gait training by the author. The participants completed gait training until they were comfortable ambulating without assistance or walking aids. Data collection was completed at Holland Bloorview during a one-hour session.

During data collection, participants completed four walking trials on level ground. Level ground was selected because the participants were new users of the AT-Knee, and were not experienced
enough to walk on other terrain (i.e., stairs or incline). Each trial was 7 m in length, and participants were instructed to walk at a self-selected speed.

6.1.2.3 Measuring WGAT Function

6.1.2.3.1 Data Synchronization and Step Extraction

Data was synchronized similar to Andrysek et al by detecting the first instance of changes in FZ < -30000 N/s and changes in Acc vertical > 40 bits/s which corresponded to a stomp [26], [34]. If the algorithm could not detect stomp peak, or selected an incorrect timestamp for the stomp, the trials were synchronized manually.

Only steady state steps were extracted. For each participant, 5-10 steps were extracted for analysis, using the threshold of FZ (0-30 N) to detect 5-10 instances of stance phase. Only the stance phase portion of each step was extracted for analysis, similar to Andrysek et al [34].

6.1.2.3.2 Comparison of Extension and Flexion Detection

6.1.2.3.2.1 Comparison of KA Moment to Angled LPS and Force-Sensing Resistor

Thresholds of the knee axis moment were used as described by Andrysek et al to determine if the moment was an extension or a flexion moment [34]. Similarly, thresholds were determined for the angled LPS and the force-sensing resistor to determine if the knee was extended or flexed. The selection of thresholds used to define extension and flexion for the FSR and angled LPS was done by trial and error by iteratively analyzing the signals and maximizing the overall accuracy in detection. Thresholds for each participant were then recorded. The detection of correct extension and flexion occurred if the sensor of the WGAT made the same detection as the KA moment. Contrarily, the WGAT sensors made an incorrect detection if they detected the opposite of the KA moment (i.e. if they detected extension when the KA moment detected flexion). The accuracy of the sensors were calculated using Equation 8 for each step, and then the mean and standard deviation across all steps was calculated.

\[
\text{accuracy} = \frac{\text{number of events of correct extension} + \text{number of events of correct flexion}}{\text{total events}} \times 100 \quad (8)
\]
6.1.2.3.2.2 Comparison of CA Moment to Lock Position

Similarly to Section 6.1.2.3.2, thresholds of the control axis moment were used as described by Andrysek et al to determine if the moment was an extension or a flexion moment [34]. Thresholds were also determined for the LPS used to measure lock position, to determine if the knee was locked or unlocked. The default threshold was 0.09 mm as described by Andrysek et al [34], but if this did not maximize the accuracy, the threshold was determined by iterative trial and error. The detection of correct extension and flexion occurred if the LPS made the same detection as the CA moment. Contrarily, the LPS made an incorrect detection if it detected the opposite of the CA moment (i.e. if it detected extension when the CA moment detected flexion). Equation 8 was also used to calculate the accuracy of the LPS per step, and then the mean and standard deviation were calculated.

6.1.2.3.3 Comparing WGAT Function to Load Transducer

After determining the accuracy and thresholds for the three sensors of the WGAT, a comparison in state detection by the WGAT and load transducer was completed. Because this project measured WGAT function as compared to the load transducer (as used by Andrysek et al), the analysis strategy mimicked what was done by Andrysek et al [34]. Performance of the load transducer system was determined as described in 2.3.2.2.

Two sensors in the WGAT were used to determine the states of the system: the force-sensing resistor and the LPS used to measure lock position. The binary states of the two input variables were used to determine the controller state (see Table 7), and then occurrences of states as percentages of stance phase were calculated.

More information about the calculation and determination of states can be found in the MATLAB code in Appendix C.
Table 7: 4 finite control states, based on the binary detection of extension and flexion by the FSR and LPS.

<table>
<thead>
<tr>
<th>Control State</th>
<th>FSR</th>
<th>LPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ext</td>
<td>Flex</td>
</tr>
<tr>
<td>A</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>D</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

6.2 Results

6.2.1 Comparison of Extension and Flexion Detection

The percentage accuracies of the sensors of the WGAT in detecting extension and flexion were calculated by summing the binary instances of correct detection and dividing by the total number of events. This gave a percentage accuracy out of 100%, compared to the gold standard load cell. Further, the mean difference in detection of flexion was calculated to determine whether any decrease in accuracy was due to over- or under-detection of flexion.

Table 8 summarizes the thresholds determined for each participant that maximized overall accuracy of the system. Values above the FSR and angled LPS thresholds indicated extension, and values above the lock position threshold indicated the lock was unlocked which corresponded to an extension moment at the control axis. Table 8 also shows whether the participant’s data automatically synced, as described in 6.1.2.3.1, or if the data was manually synced by the author.

Table 8: Thresholds for determination of extension and flexion for FSR, angled LPS and lock position. Further, synchronization type for each participant is displayed.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Threshold</th>
<th>Synchronization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FSR</td>
<td>Angled LPS</td>
</tr>
<tr>
<td>1</td>
<td>300 bits</td>
<td>-0.4 mm</td>
</tr>
<tr>
<td>2</td>
<td>300 bits</td>
<td>-0.09 mm</td>
</tr>
<tr>
<td>3</td>
<td>250 bits</td>
<td>-0.2 mm</td>
</tr>
<tr>
<td>4</td>
<td>400 bits</td>
<td>-0.2 mm</td>
</tr>
<tr>
<td>5</td>
<td>500 bits</td>
<td>-0.2 mm</td>
</tr>
<tr>
<td>6</td>
<td>300 bits</td>
<td>-0.2 mm</td>
</tr>
</tbody>
</table>
6.2.1.1 Comparison of KA Moment to Angled LPS

Detection of extension and flexion were compared for the load cell (measuring KA moment) and the angled LPS. Figure 24 shows two plots comparing KA moment and the angled LPS as example data for the comparison carried out. The left graph shows the participant with the highest accuracy, and the graph on the right shows the participant with the lowest accuracy.

Figure 24: Example data for comparison of KA moment to angled LPS. Figure includes measurement of KA moment (units: Nm; convention: extension moment positive) and angled LPS (units: mm X 5; convention: flexion negative). Each of the 4 states (correct detection of extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (left) Participant 1AB and (right) Participant 3AB.

Accuracy of the angled LPS compared to gold standard ranged from 74.6% to 92.9% (mean 83.1% and SD 6.50%). Overall, the angled LPS showed to have an over-detection of flexion. More detailed information about the mean difference in detection of flexion for each participant can be seen in Table 24.
Table 9: Accuracy of the angled LPS in detecting extension/flexion compared to the KA moment, for all participants.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Accuracy (%)</th>
<th>SD (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1AB</td>
<td>92.9</td>
<td>3.63</td>
</tr>
<tr>
<td>2AB</td>
<td>82.7</td>
<td>7.97</td>
</tr>
<tr>
<td>3AB</td>
<td>74.6</td>
<td>6.76</td>
</tr>
<tr>
<td>4AB</td>
<td>84.1</td>
<td>3.85</td>
</tr>
<tr>
<td>5AB</td>
<td>77.7</td>
<td>2.98</td>
</tr>
<tr>
<td>6AB</td>
<td>86.7</td>
<td>6.07</td>
</tr>
<tr>
<td>All</td>
<td>83.1</td>
<td>6.50</td>
</tr>
</tbody>
</table>

6.2.1.2 Comparison of KA Moment to Force-Sensing Resistor

Similarly to Section 6.2.1.1, detection of extension and flexion were compared for KA moment as measured by the load cell and the FSR. Figure 25 shows two plots comparing KA moment and the FSR as example data for the comparison carried out. The left graph shows the participant with the highest accuracy, and the graph on the right shows the participant with the lowest accuracy.

Figure 25: Example data for comparison of KA moment to the force-sensing resistor. Figure includes measurement of KA moment (units: Nm; convention: extension moment positive) and force (units: bits/50; convention: extension positive). Each of the 4 states (correct detection of
extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (left) Participant 1AB and (right) Participant 3AB.

Relatedly to the angled LPS, participant 1AB had the highest accuracy at 93.0%. Overall, the accuracy of the FSR ranged from 82.6% to 93.0% (mean 87.4% and SD 3.77%). The FSR also over-detected flexion, as can be seen in Table 24.

Table 10: Accuracy of the force-sensing resistor in detecting extension/flexion compared to the KA moment, for all participants.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Accuracy (%)</th>
<th>SD (%)</th>
</tr>
</thead>
</table>
| 1AB         | 93.0         | 4.08
| 2AB         | 88.2         | 2.98
| 3AB         | 83.7         | 4.37
| 4AB         | 88.9         | 6.22
| 5AB         | 82.6         | 3.71
| 6AB         | 87.7         | 6.69
| All         | 87.4         | 3.77 |

6.2.1.3 Comparison of CA Moment to Lock Position

The last comparison was detection of extension and flexion from the CA moment as measured by the load cell and the LPS. Figure 25 shows two plots comparing CA moment and the LPS as example data for the comparison carried out. The left graph shows participant 1AB and the right graph shows participant 4AB.
**Figure 26:** Example of data for comparison of CA moment to lock position. Figure includes measurement of CA moment (units: Nm; convention: extension moment positive) and LPS (units: mm X 10; convention: locked < threshold, unlocked > threshold). Each of the 4 states (correct detection of extension/flexion, and incorrect detection of extension/flexion) are depicted at the bottom as on (high) or off (low). (*left*) Participant 1AB and (*right*) Participant 4AB.

Contrarily to what was seen with the angled LPS and FSR, when comparing the CA moment to lock position, the LPS had a high accuracy (mean 96.8% and SD 1.13%). Table 11 shows the overall accuracies and standard deviations for each participant.

**Table 11:** Accuracy of the LPS in detecting extension/flexion compared to the CA moment, for all participants.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Mean Accuracy (%)</th>
<th>SD (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1AB</td>
<td>97.5</td>
<td>2.19</td>
</tr>
<tr>
<td>2AB</td>
<td>97.3</td>
<td>1.81</td>
</tr>
<tr>
<td>3AB</td>
<td>97.2</td>
<td>2.59</td>
</tr>
<tr>
<td>4AB</td>
<td>94.6</td>
<td>3.92</td>
</tr>
<tr>
<td>5AB</td>
<td>97.6</td>
<td>1.01</td>
</tr>
<tr>
<td>6AB</td>
<td>96.9</td>
<td>2.76</td>
</tr>
<tr>
<td><strong>All</strong></td>
<td><strong>96.8</strong></td>
<td><strong>1.13</strong></td>
</tr>
</tbody>
</table>

**6.2.2 State Analysis**

Based on the accuracies calculated in Section 6.2.1, 4 finite output states were created using the LPS detecting lock position (as it was shown to accurately replace the load transducer in detecting CA moment extension/flexion), and the FSR. The FSR was chosen over the angled LPS because it had a higher detection accuracy.

From the paper by Andrysek et al, they stated that according to their 8-state model, the expected controller states were states 2, 3, 6, and 7 [34]. These states correspond to a knee extension moment (state 2), mid-stance with lock engaged and knee under stable loading (state 3), controller is free to flex (i.e. pre-swing) (state 6), knee is locked and providing stability (state 7) [34]. Table 12 was created to correlate the 4 finite output states with the expected states of the 8-state model. This correlation was done by mapping the KA moment variable to the FSR, and mapping the CA moment and lock position variables to the LPS. In addition, unexpected states were determined as possible representations of states A to D, due to the fact that the CA moment
and lock position variables were being combined into a single variable. As the LPS detecting lock position was shown to accurately represent the CA moment (i.e. flexion moment corresponded to locked, extension moment corresponded to unlocked), unexpected states that satisfied one of the variables but not the other were deemed to be possible unexpected states that could be detected in addition to the expected state. The combination of the expected state and one of the unexpected states was used to determine if information from two states was being absorbed into a single state.

**Table 12:** Conditions of 4 finite output states compared to 8-state model, based on using CA moment as determination of expected and unexpected states.

<table>
<thead>
<tr>
<th>Control State</th>
<th>FSR</th>
<th>Lock</th>
<th>8-State Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ext</td>
<td>Flex</td>
<td>Expected State</td>
</tr>
<tr>
<td>A</td>
<td>X</td>
<td>X</td>
<td>3 (mid-stance)</td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>X</td>
<td>7 (heel strike)</td>
</tr>
<tr>
<td>C</td>
<td>X</td>
<td>X</td>
<td>2 (late stance)</td>
</tr>
<tr>
<td>D</td>
<td>X</td>
<td>X</td>
<td>6 (toe off)</td>
</tr>
</tbody>
</table>

Figure 27 shows examples of plotted state analysis data for participants 5AB (left) and 6AB (right). Each state is represented by a different line, and depicted as on (high) or off (low). The differences between the two models were further quantified based on comparing percentage of stance phase for each state of the 4 finite output states compared to either: (a) the corresponding expected state (results in Table 13), (b) the expected state plus the unexpected state according to the CA moment (results in Table 14) or (c) the expected state plus the unexpected state according to the lock position (results in Table 15). See Figure 28 for graphs representing the states and conditions. The data was shown to be normally distributed as per Shapiro-Wilks tests, and from there, paired t-tests were performed for all three comparisons. When state D was compared to state 6 and states 5+6, p-values were less than 0.05, indicating a statistically significant difference. For all other compared conditions, p-values were greater than 0.05, indicating that there was no statistically significant difference between the states being compared.
**Figure 27:** State analysis using the load transducer compared to the WGAT during stance phase. Gait phase is delineated by the vertical blue lines, and states are depicted as on (high) or off (low). *(left)* Participant 5 and *(right)* Participant 6.

**Table 13:** Condition 1: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 3, B and 7, C and 2, and D and 6. Statistically significant differences ( \( p < 0.05 \)) are indicated by *.

<table>
<thead>
<tr>
<th>AB</th>
<th>8-state model</th>
<th>4 finite output states</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0.000</td>
<td>0.404</td>
</tr>
<tr>
<td>2</td>
<td>0.017</td>
<td>0.447</td>
</tr>
<tr>
<td>3</td>
<td>0.033</td>
<td>0.641</td>
</tr>
<tr>
<td>4</td>
<td>0.043</td>
<td>0.267</td>
</tr>
<tr>
<td>5</td>
<td>0.003</td>
<td>0.379</td>
</tr>
<tr>
<td>6</td>
<td>0.027</td>
<td>0.372</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>8-state model</th>
<th>4 finite output states</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>0.021</td>
<td>0.418</td>
</tr>
<tr>
<td>S</td>
<td>0.017</td>
<td>0.124</td>
</tr>
</tbody>
</table>

| p-value | 0.615 | 0.136 | 0.778 | 0.030* |
Table 14: Condition 2: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 3+4, B and 7+8, C and 1+2, and D and 5+6. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>8-state model</th>
<th>4 finite output states</th>
</tr>
</thead>
<tbody>
<tr>
<td>1+2</td>
<td>3+4</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>1</td>
<td>0.404</td>
</tr>
<tr>
<td>2</td>
<td>0.463</td>
</tr>
<tr>
<td>3</td>
<td>0.674</td>
</tr>
<tr>
<td>4</td>
<td>0.310</td>
</tr>
<tr>
<td>5</td>
<td>0.382</td>
</tr>
<tr>
<td>6</td>
<td>0.399</td>
</tr>
<tr>
<td>M</td>
<td>0.439</td>
</tr>
<tr>
<td>S</td>
<td>0.125</td>
</tr>
<tr>
<td>SD</td>
<td>0.042</td>
</tr>
</tbody>
</table>

p-value | 0.200 | 0.170 | 0.340 | 0.048* |

Table 15: Condition 3: percentage of gait cycle of each state for the finite states. M is the mean and SD is the standard deviation across all 6 participants. The p-value was calculated to compare states A and 1+3, B and 5+7, C and 2+4, and D and 6+8.

<table>
<thead>
<tr>
<th>8-state model</th>
<th>4 finite output states</th>
</tr>
</thead>
<tbody>
<tr>
<td>1+3</td>
<td>2+4</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>1</td>
<td>0.017</td>
</tr>
<tr>
<td>2</td>
<td>0.023</td>
</tr>
<tr>
<td>3</td>
<td>0.047</td>
</tr>
<tr>
<td>4</td>
<td>0.073</td>
</tr>
<tr>
<td>5</td>
<td>0.007</td>
</tr>
<tr>
<td>6</td>
<td>0.117</td>
</tr>
<tr>
<td>M</td>
<td>0.047</td>
</tr>
<tr>
<td>S</td>
<td>0.042</td>
</tr>
<tr>
<td>SD</td>
<td>0.042</td>
</tr>
</tbody>
</table>

p-value | 0.107 | 0.111 | 0.610 | 0.553 |
Figure 28: Plots of the mean percentage of stance phase for each state. (A) Plots of state A and states 3, 3+4 and 1+3. (B) Plots of state B and states 7, 7+8 and 5+7. (C) Plots of state C and states 2, 1+2 and 2+4. (D) Plots of state D and states 6, 5+6 and 6+8.

However, it was of interest to determine which of the three conditions correlated best with states A, B, C and D. Therefore, the difference between one of the 4 finite output states and the corresponding 8-state model condition was calculated for each participant (for example, the difference between state A and states 1, 1+3, and 3+4) (see Figure 29). From there, a repeated measures ANOVA was performed between the three groups to determine if there was any
statistically significant difference between the comparisons. The results of this analysis (which are summarized in Table 16) showed that there were significant difference between conditions for states A (p = 0.018), B (p = 0.02), C (p = 0.018), and D (p = 0.02).

Figure 29: Plots of mean difference between the 4 finite output states and the corresponding condition. Statistically significant differences from the post-hoc paired t-tests (p < 0.05) are indicated by *.
Table 16: Results from repeated measures ANOVA. The first column indicates the state of the 4 finite output states used in the calculation, and the second column indicates the state or combination of states from the 8-state model. The mean differences and p-values are in the third and fourth columns, respectively. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>4 Finite Output States</th>
<th>8-State Model</th>
<th>Absolute Mean Difference (%)</th>
<th>SD (%)</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.621</td>
<td>2.840</td>
<td>0.018*</td>
<td></td>
</tr>
<tr>
<td>(1+3)</td>
<td>2.678</td>
<td>3.345</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3+4)</td>
<td>1.343</td>
<td>2.227</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3.027</td>
<td>4.173</td>
<td>0.002*</td>
<td></td>
</tr>
<tr>
<td>(5+7)</td>
<td>2.471</td>
<td>3.133</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(7+8)</td>
<td>5.014</td>
<td>7.669</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.764</td>
<td>6.276</td>
<td>0.018*</td>
<td></td>
</tr>
<tr>
<td>(2+4)</td>
<td>1.486</td>
<td>6.685</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1+2)</td>
<td>2.820</td>
<td>6.552</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9.734</td>
<td>7.933</td>
<td>0.002*</td>
<td></td>
</tr>
<tr>
<td>(6+8)</td>
<td>1.692</td>
<td>6.526</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(5+6)</td>
<td>9.178</td>
<td>8.615</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To determine which conditions were significantly different, paired t-tests were completed to compare each possible pairing of the three conditions that correspond to 4 finite output states, as a post-hoc analysis for the ANOVA. As can be seen in Table 18 to Table 20, statistically significant differences were reported between A-3 and A-(1+3) (p = 0.032), B-7 and B-(7+8) (p = 0.012), B-(5+7) and B-(7+8) (p = 0.022), C-2 and C-(1+2) (p = 0.032), D-6 and D-(6+8) (p = 0.012), and D-(5+6) and D-(6+8) (p = 0.022).

Table 17: Results from t-tests comparing all possible combinations of the three conditions that correspond to state A, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>First comparison state</th>
<th>Mean difference with state A (%)</th>
<th>Second comparison state</th>
<th>Mean difference with state A (%)</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.621</td>
<td>(3+4)</td>
<td>1.343</td>
<td>0.078</td>
</tr>
<tr>
<td>3</td>
<td>0.621</td>
<td>(1+3)</td>
<td>2.678</td>
<td>0.032*</td>
</tr>
<tr>
<td>(3+4)</td>
<td>1.343</td>
<td>(1+3)</td>
<td>2.678</td>
<td>0.108</td>
</tr>
</tbody>
</table>
Table 18: Results from t-tests comparing all possible combinations of the three conditions that correspond to state B, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>First comparison state</th>
<th>Mean difference with state B (%)</th>
<th>Second comparison state</th>
<th>Mean difference with state B (%)</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>3.027</td>
<td>(5+7)</td>
<td>2.471</td>
<td>0.363</td>
</tr>
<tr>
<td>7</td>
<td>3.027</td>
<td>(7+8)</td>
<td>5.014</td>
<td>0.012*</td>
</tr>
<tr>
<td>(5+7)</td>
<td>2.471</td>
<td>(7+8)</td>
<td>5.014</td>
<td>0.022*</td>
</tr>
</tbody>
</table>

Table 19: Results from t-tests comparing all possible combinations of the three conditions that correspond to state C, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>First comparison state</th>
<th>Mean difference with state C (%)</th>
<th>Second comparison state</th>
<th>Mean difference with state C (%)</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.764</td>
<td>(1+2)</td>
<td>2.820</td>
<td>0.032*</td>
</tr>
<tr>
<td>2</td>
<td>0.764</td>
<td>(2+4)</td>
<td>1.486</td>
<td>0.078</td>
</tr>
<tr>
<td>(1+2)</td>
<td>2.820</td>
<td>(2+4)</td>
<td>1.486</td>
<td>0.108</td>
</tr>
</tbody>
</table>

Table 20: Results from t-tests comparing all possible combinations of the three conditions that correspond to state D, as post-hoc analysis for the ANOVA. The first column two columns indicate the conditions being compared from the 8-state model. Statistically significant differences (p < 0.05) are indicated by *.

<table>
<thead>
<tr>
<th>First comparison state</th>
<th>Mean difference with state D (%)</th>
<th>Second comparison state</th>
<th>Mean difference with state D (%)</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>9.733</td>
<td>(5+6)</td>
<td>9.178</td>
<td>0.363</td>
</tr>
<tr>
<td>6</td>
<td>9.733</td>
<td>(6+8)</td>
<td>1.692</td>
<td>0.012*</td>
</tr>
<tr>
<td>(5+6)</td>
<td>9.178</td>
<td>(6+8)</td>
<td>1.692</td>
<td>0.022*</td>
</tr>
</tbody>
</table>
6.3 Discussion

6.3.1 Measuring WGAT Function

6.3.1.1 Comparison of Extension and Flexion Detection

For all participants, it was shown that the use of lock position to predict CA extension and flexion had a high degree of accuracy. Therefore, it was shown that using the lock position was an acceptable replacement of the CA moment, and thus could be used in the model to determine extension and flexion. An interesting point to note was that all participants except 4AB used a threshold of 0.09 mm, whereas participant 4AB used a threshold of 1.5 mm. When looking at the percentage of stance phase in extension versus flexion (see Table 23 in Appendix H), 4AB spent the least amount of time in extension than any other participant. To maximize accuracy of the LPS, the threshold for unlocked/locked detection was moved to match the amount of time in extension detected by CA moment and LPS. Therefore, it is possible that because the amount of time in extension was less, the threshold had to be increased match the reduced extension period. Participant 4AB was also the shortest participant (152 cm), and although the gait simulator was adjusted to its lowest possible height, there is the possibility that the simulator was too tall for the participant. During data collection, the author observed that 4AB took short, tentative strides and had a more difficult time disengaging the knee lock. It is possible this contributed to the increased percentage of stance phase spent with a flexion moment and the need to change the lock position threshold.

Contrarily, both the angled LPS and FSR had lower accuracies when compared to the KA moment. The FSR was more accurate than the angled LPS (87.4% mean and SD 3.77%; 83.1% mean and SD 6.50%), so in terms of choosing which sensor would best replace measurements from the load transducer, the FSR was chosen. Precision of wearable pressure sensors in gait analysis studies have shown to have accuracies ranging from 88% upwards to 96% [28], [74]. FSRs are often reported to measure the ground reaction force by being placed under the shoe, so when determining an acceptable accuracy for the FSR in this study, the range of 88-96% accuracy is an approximate guide. Although the mean accuracy of the FSR was not within the range of 88-96%, it was chosen to be used for the determination of the 4 finite output states because it had a higher accuracy than the angled LPS and its mean accuracy was only 0.6% lower than the minimum of the range.
Another interesting note is that while the accuracy of the FSR appears high, the overall
calculation of accuracy fails to show the degree to which the sensor over and under detects
flexion/extension. Looking at Table 24, both the angled LPS and FSR under-detected extension,
and over-detected flexion. While this can be attributed to the extension/flexion thresholds
selected during analysis, these thresholds were selected to maximize overall percentage
accuracy. If thresholds were changed to minimize the over-detection of flexion, this would likely
result in a reduced overall percentage accuracy.

6.3.1.2 State Analysis

In performing paired t-tests between the resulting states from the load transducer and WGAT
(see Table 13, Table 14 and Table 15), the goal was to determine if there was a statistically
significant difference between the states of the two data collection systems. This was to
determine if states A to D were different from the 8-state model. For the p-values greater than
0.05, it can be concluded that there was not a statistically significant difference between the
detection of states for any of the compared conditions. However, for state D, it was shown that
there was a statistically significant difference between D and state 6, and D and states 5+6.

However, it was also of interest to determine which combination of states from the 8-state model
was best represented by the 4 finite output states. When looking at which of the states from the 4
finite output states corresponded best to the 8-state model, the combinations that had the lowest
absolute mean difference were states A and 3, B and 5+7, C and 2, and D and 6+8. However,
state 5 is not readily explained based on the mechanics of the ASPL controller, and thus the
detection state 5 was likely due to error in the data collection system [34]. Therefore, excluding
5+7 for consideration in this analysis, state B corresponds best with state 7 of the 8-state model.

When moving from 8 to 4 states, it was hypothesized that there was some degree of information
loss as this likely results in the combination of states or exclusion of states. In attempt to better
understand this, the percentage of stance phase captured in the new combination of states was
calculated, and is presented in Table 22 of Appendix G. If only the four expected states of the 8-
state model are captured (states 2, 3, 6 and 7), the percentage of stance phase classified by the
states is between 82.7% and 92.4% (mean 88.6% and SD 4.08%). However, when considering
the combination of states 6+8 are a more accurate representation of state 4 because the absolute
mean difference is the smallest, the percentage of stance phase classified by states 2, 3, 6, 7, and
8 increases to between 92.3% and 100% (mean 96.7% and SD 2.86%). Therefore, the 4 finite output states captures a higher percentage of data from the 8-state model when using states 2, 3, 6, 7 and 8. However, in combining states 6+8, there is no delineation in the 4 output states between a state that has an extension moment at the control axis while the knee is unlocked, versus a state that has a flexion moment at the control axis while the knee is unlocked. According to Andrysek et al, state 8 can occur when there is slight knee flexion in the presence of an unlocked knee, but the occurrence of this state is expected to be low, especially because this is an unexpected state based on the design of the model [34]. Further, because of the lower accuracy of the FSR, there is the possibility that the reason combining state 6+8 resulted in better correlation with state D is because the FSR was over-detecting flexion. As this was shown in the comparison between the FSR and load transducer, it makes sense that there is a higher occurrence of state D compared to state 6 in the 8-state model. If the FSR were to detect less flexion, the occurrence of state D would decrease and better match state 6. This would likely result in a higher occurrence of state C because there would be an increase in detection of extension by the FSR in combination with the LPS detecting an unlocked position.

In applying the WGAT and 4 finite output states, researchers and designers will be able to instrument participants with the WGAT and collect data about the performance of the AT-Knee. By making perturbations to a computational model using the 4 finite output states, designers can determine if the output is an expected response, and whether the AT-Knee is performing as expected. One example would be if the friction is changed at the KA axes, hindering knee flexion, it would be expected to see a decreased occurrence of state D (toe off).

Further, although the WGAT was created for use in the AT-Knee in this study, there is future applicability to other prosthetic knees. The sensors of the WGAT could be changed to better match the parameters of interest for other prosthetic knees, and in the same way, finite output states and a computational model defined. Then, this type of study protocol could be repeated to determine the performance of other prosthetic knees.

The ultimate goal of the state analysis was to determine if the WGAT accurately classified the states of the ASPL according to a 4 finite output states, compared to the 8-state model. It is concluded that the 4 finite output states best represent states 2, 3, 6, 7, and 8, but with no ability to delineate states 6 and 8. Moreover, in using the new instrumentation and 4 finite output states,
this project allows for a portable system to be used in data collection, without the reliance on a load transducer.
Chapter 7

Limitations and Future Work

While the recruitment of able-bodied participants was convenient and gait simulators have been used in other able-body studies, there are limitations to using able-bodied participants with gait simulators. Firstly, because all the participants were inexperienced walking with the AT-Knee, only level ground could be used as a testing condition. The eventual goal would be to use the WGAT to collect data while participants walk on other terrain conditions, but the WGAT could not be validated on other terrain conditions because the study population could not walk on that terrain. Therefore, future work should be done to collect data with experienced users walking with the WGAT. In the study by Andrysek et al, the authors showed a similar accuracy of their model with the amputee participant and the able-bodied participants in their study [34]. However, results from the amputee participant with Andrysek et al showed a different mean percentage of stance phase for the different states as compared to simulated gait with the amputees, so it would be expected that if the WGAT was instrumented on an amputee participant, the mean percentage of stance phase across the 4 states would be different than with the able-bodied participants presented here [34]. Further, Andrysek et al had the ability to quantify the mean percentage of stance phase of the different states under a variety of mobility conditions, and found that the percentage of stance phase of each state changed across the different mobility conditions [34]. This would be valuable future analysis with the WGAT and 4 finite output states as it is predicted that similar results will be found. Moreover, participants were fitted with the gait simulator by an amateur who received minimal prosthetic alignment and gait training. The simulator socket was a single size, so although a trained prosthetist designed inserts to allow for adjustment to the diameter of the socket, the simulator provided limited fit options for the various participants, which could have created variability in the data collected, depending on how well the setup fit that particular participant. Moreover, even though adjustable pylons were used to modify the height of the setup, there was a minimum height for the gait simulator, and it is possible this minimum height was too tall for the shortest participants. A simulator that is too tall becomes somewhat of a pole for the participant to vault over, potentially making them more uncertain when walking and altering their normal gait.
Most trials were synchronized using automatic detection of vertical force and vertical acceleration thresholds based on stomp data. However, there was variation in the signals generated because of variability in the strength and stomp technique between participants. Therefore, in the cases where the automatic thresholds detected the wrong time stamp or could not detect the stomp, the trials had to be manually synced by the author. This could have created an error in the synchronization between the two systems because the manual detection of stomp data was done by trial and error to define the thresholds for those trials.

Due to the low sample size of this study, it’s possible that the statistical analysis performed was underpowered. Moreover, in the application of paired t-tests and repeated measures ANOVA to show there was not a significant difference between the conditions being compared, there was the possibility of over-representing Type II errors (false negatives). Although multiple comparisons were performed, corrections for multiple comparisons (i.e. Bonferroni correction) were not performed for two reasons. Firstly, the probability of identifying a significant result increases as the sample size increases, but as mentioned, the sample size for this study was low. Secondly, the multiple comparisons correction is used to reduce the chances of obtaining a false positive result (Type I error), which would increase the likelihood of over-representing Type II errors in this study. In the interest of not increasing the likelihood of Type II errors even further, corrections for multiple comparisons were not performed.

Because the WGAT mount was split into two parts (a 3D-printed mount for the sensors attached to the knee and a case attached to the thigh to hold the associated electronics), longer wires had to be used between the two parts to minimize the chance the wires will disconnect when the knee bends. However, this still happened, as was the case for the excluded participant. This is a limitation of the two part mount, and should be remedied in future iterations of the design. A single unit design would eliminate this issue, as no wires would have to leave the single box. Another potential design for exploration would be a box that attaches to the pylon instead of the thigh, because the wires would not require extra slack to withstand the bending of the knee.

Further, there was a large dependence on the load transducer in this project. While the load transducer’s main purpose was to validate the WGAT, its was required to determine the thresholds for the LPSs and FSR. This is a limitation because the final WGAT would not have the load transducer; the KA and CA moments would not be present to compare extension/flexion
with and determine the best thresholds for the WGAT to maximize accuracy. A sensitivity analysis was performed to better quantify the effect the threshold had on the mean accuracy across all participants (see Appendix I). As compared to the mean accuracy determined based on individual thresholds (83.1% for the angled LPS; 87.4% for the FSR), the accuracy determined by an average threshold yielded a lower value for both sensors (78.2% for the angled LPS; 83.4% for the FSR). Future work could involve collecting data from more participants to determine the best average threshold across a larger population. Additionally, the FZ measurement from the load transducer was also used in stance detection. In future iterations of the WGAT and 4 finite output states, stance detection should be done with other sensors, such as the accelerometer, as literature has shown that accelerometers can successfully detect gait events [74]–[83].

Although the load transducer is a gold standard, the calculation of KA moment in this project is based on assumptions and calculations, transforming the measurements from the load transducer axis to the knee axis. This study chose to compare the measurement of the FSR to the load cell, but it is possible that some of the differences in the accuracies result from inaccuracies in the load transducer and 8-state model. Since the FSR directly measures in the knee axis, it is possible that the FSR actually provides a more accurate representation of flexion and extension at KA. Further studies could confirm accuracy of the WGAT using other gold standard methods, such as video analysis.

In comparing the mean percentage of stance phase for the various states, the analysis did not account for any differences in period or timing of the different states (i.e. if there was a delay in the beginning of a state). However, in addition to the analysis presented in 6.2.2, the graphs of the output states were analyzed visually for any significant timing offsets in the initiation or termination of states. While this visual analysis showed that the states generally aligned between the 4 and 8 output states, future work should involve analyzing the periods of time that the states are present, and providing correlation between these periods of time.

Finally, a limitation of the state analysis is that the 4 finite output states were developed using the FSR which was shown to detect extension and flexion with only moderately high accuracy. There is a need to better represent the detection of extension/flexion with respect to the knee axis. FSRs are known to be inconsistent in their measurements, and subject to wear and tear.
which affects the output of the sensor [74]. Moreover, movement of the sensor placement causes differences in the output, which reduces the accuracy. Future work could be done to use the FSR in a more effective way, perhaps by changing the thicknesses of the rubber buffer to determine if a thicker or thinner buffer is more effective, as only a single sized rubber buffer was used in this project. Alternatively, the angle of the LPS could be changed to detect a different range of movement of the thigh portion of the knee. Further, a different sensor altogether could be explored to replace these two sensors of the WGAT.
Chapter 8

8 Conclusion

The development of the wearable gait analysis tool presented in this thesis addressed the need to further develop a sensor system to monitor the internal function of the AT-Knee. This was done through the development of a 3D-printed mount that easily attached to and detached from the knee, and did not require the disassembly of the prosthetic knee, pylon or foot. Moreover, replacing the inductive proximity sensor with a linear position sensor reduced the weight and size of the overall system by integrating a smaller sensor that had a lower power requirement.

The results from this study showed that the LPS could accurately replace the load transducer in determining extension and flexion moments about the control axis. Further, the results showed that the combination of the FSR and LPS to delineate 4 finite output states could replace the load transducer and 8-state model in detecting binary states of the ASPL controller.

With the addition of a finalized user interface, and more work to accurately detect extension and flexion about the knee axis (either by changing the detection thresholds with the FSR or angled LPS, or by using an additional sensor), the WGAT has the potential to be used in larger studies to inform prosthetic knee design, especially by informing the performance of the knee under a variety of environmental conditions due to the portability of the system.

The main perceived contributions to the field are:

1. Research into the need for a wearable gait analysis tool that inputs empirical data into a computational model, to better inform the engineering design process;

2. The development of a fully wearable gait analysis tool;

3. The development of 4 finite output states based on the binary conditions of 2 input variables;

4. The validation of the WGAT and 4 finite output states through collection of data from walking trials with multiple able-bodied participants.
References


[57] M. Arnout, C. Pierre, V. D. Michael, V. Bram, and L. Dirk, “Concept and design of the HEKTA (Harvest Energy from the Knee and Transfer it to the Ankle) transfemoral


Appendices

Appendix A: Arduino Code

/* Arduino code to control the wearable analysis tool
 *
 */

// Include header files for SPI and SD card
#include <SPI.h>
#include <SD.h>
#include <Wire.h>
#include <Time.h>
#include <TimeLib.h>

// these constants describe the pins. They won't change:
const int groundpin = 18; // analog input pin 4 -- ground
const int powerpin = 19; // analog input pin 5 -- voltage
const int FSR = A0; // flexiforce reading
const int LPS_position = A4; // linear position from sensor
const int LPS_angle = A3; // reading of knee articulation from LPS
//const int xpin = A2; // accelerometer x-axis
const int ypin = A2; // accelerometer y-axis
const int zpin = A1; // accelerometer z-axis (only on 3-axis models)
const int analogPin;

//digitalRead/Write pins
const int switchPin = 3;
const int LEDPin = 5;

//The min and max values from accelerometer while standing still
int minVal = 257;
int maxVal = 393;
//The min and max values from LPS
int minPVal = 0;
int maxPVal = 1023;

String dataString; // will append data to this string

File myFile;

void setup() {
  pinMode(switchPin, INPUT); //initializes switch pin as an input
  pinMode(LEDPin, OUTPUT); //initializes LED pin as output
// initialize the serial communications:
Serial.begin(9600);
pinMode(10, OUTPUT);
digitalWrite(10, HIGH);

Serial.begin(9600);
while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
}
Serial.print("Initializing SD card...");
if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    while (1);
}
Serial.println("initialization done.");
File myFile = SD.open("trial.txt",FILE_WRITE); //open file to write to it
myFile.println("New Test"); //separate data
myFile.close(); //close the file
}

void loop() {
    File myFile = SD.open("trial.txt",FILE_WRITE); //open file to write to it
    if(myFile){ //if file opens,
        digitalWrite(LEDPin, HIGH); //turn LED on
        // Reset data string
        dataString = "";
        for(int analogPin = 0;analogPin < 5;analogPin ++){ //read sensor analog pins
            int sensor = analogRead(analogPin);
            dataString += String(sensor);
            dataString += ",";
        }
        myFile.println(dataString); //print dataString in file
        Serial.println(dataString); //print dataString (time and sensor values) to serial monitor
    }
    else{ //if file does not open
        Serial.println("Error in opening the file."); return;
    }
    myFile.close(); //close the file when switch is OFF
}
Appendix B: MATLAB Code for Extension/Flexion Detection Comparison

function [Counter] = engval(Counter, WriteData, StepInputs) %call function by' [Counter]=sensorState(Counter, WriteData, StepInputs)'
close
%Counter is used for writing to lines and sheets in excel - set to 1
%Write data: 1 - Writes full data to Excel - SLOW; 0 - Only writes summary %data - FAST PROCESSING
%StepInputs: 1 - Prompts, 0 - takes data from Excel file cells J1 and K1
%where it has been
%previously manually inputted. If no data was inputted, set StepInput to 1

% MODEL INPUTS set these values based on model % makes adjustment for lock spring based on lock position and hence lock spring contribution (i.e. 3Nm when fully unlocked) and friction measured to be 4Nm @ full knee extension
KASpringMoment = 1; % (1 Nm for ext assist, a
KASpringMoment = KASpringMoment - 0; % (1Nm, 2kg x 50mm for secondary ext spring
CASpringMoment = 2; % was measured at 2Nm
KAFrictionMoment = 0; % was measured at 2Nm
CAFrictionMoment = 0; % was measured at 0.5Nm
Unlocked = 0.9; % position of unlocked lock (1 mm typical)
Locked = 0.9; % position of locked (1 mm to 0.5 when more securely locked)

% Processes and plots IPS, FSR, Accelerometer, and load transducer data
% Synchronizes ASPL-SS data with ATI data based on slope change within first % 500 readings for AccX and FZ which correspond to stomp action

% Load all system data
%[atiFile,atiPath] = uigetfile('*.csv','Select the ATI data file.');
%[asplssFile,asplssPath] = uigetfile('*.csv','Select the ASPL-SS data file.');

% Load ATI .csv file
atiPath = '/Users/rachelreding/Documents/2016-2018 Clin Eng/Thesis Files/11-ATI/1/trial2';
asplssPath = atiPath;
asplssFile = 'test-1-june25-t2.csv';

%[atiFile,atiPath] = uigetfile('*.csv','Select the ATI data file.');
atiData = csvread(fullfile(atiPath,atiFile));

% Load ASPLSS .csv file
%[asplssFile,asplssPath] = uigetfile('*.csv','Select the ASPLSS data file.');
asplssData = csvread(fullfile(asplssPath,asplssFile));

% Time stamps
% Calculate seconds since epoch day (00:00:00, 1st January 1990) to date
% and time for ATI data based on file name and Excel timestamps
dateA= regexp(atiFile, '(?<=\b\[(\d{4})(\-(\d{2})(\-(\d{2})(\-(\d{2}))\)?\])\b', 'match'); % regexp extracts % date and time string from ATI filename
% Process date and time string to a format readable by MATLAB
dateA{1}(1:3)=''; % replaces day (columns 1-3) with '
dateB=strrep(dateA{1},'EDT','');
dateC=strrep(dateB,'-',':');
ref=datevec('01-01-1900, 00:00:00'); %Transform epoch day into date vector
cur=datevec(dateC); %Transform current time into date vector
NTPTime=etime(cur,ref); %Elapsed secs since epoch day/time to current day/time
cur2=cur;
cur2(1,4:6)=0;
NTPTime2=etime(cur2,ref); %Elapsed secs since epoch day/time to current day

% ATI
ModNTPTime = floor(NTPTime/(2^20))*(2^20); % Masks lower 20 bits of NTP Time
CorNTPTime = ModNTPTime-10800; %Corrects for EDT Timezone
ROTime = atiData(:,1)/4096; %Seconds since last rollover
ROTime(ROTime<0) = ROTime(ROTime<0) + 2^20; % Make -ve timestamps +ve
atiTime = ROTime+CorNTPTime; %col vec = # secs since epoch time for each row
atiData(:,1) = atiTime; %substitute col vec for original ATI time stamps

% % Synchronization

% Use if manually sync'ed
% ATI Fz
fzStartInd = 1;
atiTime = atiData(:,1)-atiData(fzStartInd,1); %set sync timestamp = 0

% ASPLSS AccX
accZStartInd = 1;

asplssTime = asplssData(:,6)-double(asplssData(accZStartInd,6)); %set sync timestamp = 0
asplssTime = asplssTime/1000.0; %convert to seconds from ms

% Script for synchronization
% ATI Fz
atiDataTrunc = atiData(1:2000,9); %create a truncated matrix of ATI data
fzDiff = 0;
i = 1;
while (fzDiff) > -30000 %search for first instance of deltaFz > -30000
    i = i+1;
    fzDiff = atiDataTrunc(i)-atiDataTrunc(i-1);
end
fzStartInd = i-1;
atiTime = atiData(:,1)-atiData(fzStartInd,1); %set sync timestamp = 0

% ASPLSS AccZ
asplssDataTrunc = asplssData(1:500,2); %create a truncated matrix of ASPL-SS data
accZDiff = 0;
j = 1;
while (accZDiff) < 40 %search for first instance of deltaAccZ > 40
    j = j+1;
    accZDiff = asplssDataTrunc(j)-asplssDataTrunc(j-1);
end
accZStartInd = j-1;
asplssTime = asplssData(:,6)-double(asplssData(accZStartInd,6)); %set sync timestamp = 0
asplssTime = asplssTime/1000.0; %convert to seconds from ms
%% neutralProx - Calibrates lock position measurement
neutralProx = max(asplssData(:,4));

%% neutralAng - Calibrates measurement from angled LPS
neutralAng = max(asplssData(:,5));

%% Data set variables

AccX = asplssData(:,2)/100; %Vertical acceleration
AccZ = asplssData(:,3)/100; %Horizontal acceleration
Prox = -((asplssData(:,4)-neutralProx)/80.55); %Lock position (mm)
Pres = asplssData(:,1); %FSR force
Ang = (asplssData(:,5)-neutralAng)/80.55; %angled sensor

% Cut off data before 0 timestamp
AccX=AccX(accZStartInd:length(asplssTime));
AccZ=AccZ(accZStartInd:length(asplssTime));
Prox=Prox(accZStartInd:length(asplssTime));
Pres=Pres(accZStartInd:length(asplssTime));
Ang=Ang(accZStartInd:length(asplssTime));

FX = atiData(:,7)/1000; %/1000 to correct for ATI errors
FY = atiData(:,8)/1000; %/1000 to correct for ATI errors
FZ = atiData(:,9)/1000; %/1000 to correct for ATI errors
TY = atiData(:,11)/1000; %/1000 to correct for ATI errors
DX = 0.0235;
DXZ = 0.02; % in flexion with lock in locked position
DZ = 0.06433;
DZ2 = 0.1953;
Theta = 0;

% Cut off data before 0 timestamp
FX=FX(fzStartInd:length(atiTime));
FY=FY(fzStartInd:length(atiTime));
FZ=FZ(fzStartInd:length(atiTime));
TY=TY(fzStartInd:length(atiTime));

%% Synchronize start times to the 0 time point set
x1=asplssTime;
x2=atiTime;
minX1 = min(x1);
maxX1 = max(x1);
x1Range=minX1+maxX1;

x1=x1(accZStartInd:length(x1));

%x1 = x1-minX1;

minX2 = min(x2);
maxX2 = max(x2);
x2Range=minX2+maxX2;

%x2 = x2-minX2;
x2=x2(fzStartInd:length(x2));

figure
plot(x1,Prox);
hold on
plot(x1,AccX);
plot(x2,-FZ/100);
hold off

%% Interpolate values for Prox
x1Range= 0:0.01:x1Range;
x1Range=transpose(x1Range);
Prox=interp1(x1, Prox, x1Range, 'spline');
Ang=interp1(x1, Ang, x1Range, 'spline');
AccX=interp1(x1, AccX, x1Range, 'spline');
Pres=interp1(x1, Pres, x1Range, 'spline');
AccZ=interp1(x1, AccZ, x1Range, 'spline');

%% Interpolate values for ATI data
x2Range= 0:0.01:x2Range;
x2Range=transpose(x2Range);
FX=interp1(x2, FX, x2Range, 'spline');
FY=interp1(x2, FY, x2Range, 'spline');
FZ=interp1(x2, FZ, x2Range, 'spline');
TY=interp1(x2, TY, x2Range, 'spline');

%% Cut off ATI and ASPLSS data at same end point
% find minimum end time stamp between the two sensor systems
maxX= min(max(x2Range),max(x1Range));

% shorten xRanges to minimum end time between two systems
i=1;
while x1Range(i) ~= maxX
    i=i+1;
end

x1Range=x1Range(1:i);
% x2Range=x2Range(1:j);
xRange=x1Range;

% shorten data sets to minimum end time between two systems
Prox=Prox(1:i);
AccX=AccX(1:i);
AccZ=AccZ(1:i);
Ang=Ang(1:i);
Pres=Pres(1:i);

FX=FX(1:i);
FY=FY(1:i);
FZ=FZ(1:i);
Fz=FZ(1:i);
TY=TY(1:i);

%% ATI data origin asdjustment
DX2=DX2+(Prox*2)/1000;  % compensate for change in location of KA from loadcell due to movement at CA
DX2length= length(DX2);
CAMoment = (TY*cos(Theta)) + (FX*cos(Theta))*DZ - (FY*sin(Theta))*DZ + FZ*DX;  
%control axis
KAMoment = (TY*cos(Theta)) + (FX*cos(Theta))*DZ2 - (FY*sin(Theta))*DZ2 + FZ.*DX2;  %knee axis

%% Remove non steps prior to first step
figure
[h1]=plot(xRange,Fz,xRange,(Fz/100)+20);
axis([0 inf 0 100])
grid on;
if StepInputs==1
%User-entered StartTime data start (s)
prompt = {'Start time of steps'};
dlg_title = 'Start time';
num_lines = 1;
def = {'4'};
answer = inputdlg(prompt,dlg_title,num_lines,def);
StartTime = str2double(answer{1})*100;

%Set Fz value to be smaller (sinc Fz is negative) than the smallest value
%occurring during swing-phase
%Fz is negative) swing-phase values
prompt = {'Fz threshold'};
dlg_title = 'Fz threshold';
num_lines = 1;
def = {'30'};
answer = inputdlg(prompt,dlg_title,num_lines,def);
FzThreshold = str2double(answer{1});
elseif StepInputs==0;
StartTime = asplssData(1,6)*100;
FzThreshold = asplssData(1,11);
end

%%

% determine how many numbers are in xRange
LengthxRange=length(xRange);
% change xRange to go from start time, to end of range
xRange=xRange(StartTime:LengthxRange);

% change variables to show from start time to end of range
Prox=Prox(StartTime:LengthxRange);
Pres=Pres(StartTime:LengthxRange);
Ang=Ang(StartTime:LengthxRange);
KAMoment=KAMoment(StartTime:LengthxRange);
CAMoment=CAMoment(StartTime:LengthxRange);
DiffAccX=diff(AccX(StartTime-1:LengthxRange));
DiffAccZ=diff(AccZ(StartTime-1:LengthxRange));
AccX=AccX(StartTime:LengthxRange);
AccZ=AccZ(StartTime:LengthxRange);
DiffFz = Fz(StartTime-1:LengthxRange);
Fz=Fz(StartTime:LengthxRange);
DiffFz = diff(Fz);
DiffCAMomentLogic = diff(CAMoment);
DiffCAMomentLogicFlex = -1*(DiffCAMomentLogic>0);
DiffCAMomentLogicExt = (DiffCAMomentLogic<0);
DiffCAMomentLogic = DiffCAMomentLogicFlex + DiffCAMomentLogicExt;
DiffKAMomentLogic = diff(KAMoment);
DiffKAMomentLogicFlex = -1*(DiffKAMomentLogic>0);
DiffKAMomentLogicExt = (DiffKAMomentLogic<0);
DiffKAMomentLogic = DiffKAMomentLogicFlex + DiffKAMomentLogicExt;

%% Identifying gait events with ACCx and ACCz
AccXEventVert=AccX > 1.5;
AccZEventHor=AccZ>1;

%% Finite state analysis.

% Finite state analysis for all steps
KAMomentLogicExt=KAMoment>KAFrictionMoment;
KAMomentLogicFlex=KAMoment<KAFrictionMoment;
CAMomentLogicExt=CAMoment>CAFrictionMoment;
CAMomentLogicFlex=CAMoment<CAFrictionMoment;
ProxLogicUnlock=Prox>Unlocked;
ProxLogicLock=Prox<Locked;%was 0.5
PresLogicExt=Pres>500;
PresLogicFlex=Pres<500;
AngLogicExt=Ang>-0.15;
AngLogicFlex=Ang<-0.15;
FzLogic=Fz<FzThreshold;
OneArray=ones(length(xRange),1);

% Determination of states based on binary logic
State1=((KAMomentLogicExt+CAMomentLogicExt+ProxLogicLock+FzLogic)==4);
State2=((KAMomentLogicExt+CAMomentLogicExt+ProxLogicUnlock+FzLogic)==4);
State3=((KAMomentLogicExt+CAMomentLogicFlex+ProxLogicLock+FzLogic)==4);
State4=((KAMomentLogicExt+CAMomentLogicFlex+ProxLogicUnlock+FzLogic)==4);
State5=((KAMomentLogicFlex+CAMomentLogicExt+ProxLogicLock+FzLogic)==4);
State6=((KAMomentLogicFlex+CAMomentLogicExt+ProxLogicUnlock+FzLogic)==4);
State7=((KAMomentLogicFlex+CAMomentLogicFlex+ProxLogicLock+FzLogic)==4);
State8=((KAMomentLogicFlex+CAMomentLogicFlex+ProxLogicUnlock+FzLogic)==4);
StateUnd = State1 + State2 + State3 + State4 + State5 + State6 + State7 + State8;
StateUnd = ~StateUnd.*FzLogic;
StateUnd=~StateUnd==0;

%State proportion of Stance-phase
StanceLength = sum(FzLogic);
State1OnStance = sum(State1)/StanceLength;
State2OnStance = sum(State2)/StanceLength;
State3OnStance = sum(State3)/StanceLength;
State4OnStance = sum(State4)/StanceLength;
State5OnStance = sum(State5)/StanceLength;
State6OnStance = sum(State6)/StanceLength;
State7OnStance = sum(State7)/StanceLength;
State8OnStance = sum(State8)/StanceLength;
StateUOnStance = sum(StateUnd)/StanceLength;

TotalStatesOn = State1OnStance + State2OnStance + State3OnStance
+State4OnStance + State5OnStance + State6OnStance + State7OnStance;
TotalStatesOn2367 = State2OnStance + State3OnStance + State6OnStance +
State7OnStance;
TotalStatesOn234678 = State2OnStance + State3OnStance + State4OnStance +
State6OnStance + State7OnStance + State8OnStance;

TotalStates=[State1OnStance, State2OnStance, State3OnStance, State4OnStance, State5OnStance, State6OnStance, State7OnStance, State8OnStance, StateUOnStance, TotalStatesOn, TotalStatesOn2367, TotalStatesOn234678];

%For plotting the states
State1Plot=State1*3+OneArray*-50;
State2Plot=State2*3+OneArray*-55;
State3Plot=State3*3+OneArray*-60;
State4Plot=State4*3+OneArray*-65;
State5Plot=State5*3+OneArray*-70;
State6Plot=State6*3+OneArray*-75;
State7Plot=State7*3+OneArray*-80;
State8Plot=State8*3+OneArray*-85;
StateUndPlot=StateUnd*3+OneArray*-90;

% plotting of all steps for finite state analysis
h12=figure;
plot(xRange,[(FzLogic*400)-200,KAMoment, CAMoment, -Fz/10, Prox*10, Pres/50,
State1Plot, State2Plot, State3Plot, State4Plot, State5Plot, State6Plot,
State7Plot, State8Plot,StateUndPlot, 10*AccX+100, 10*AccZ+90, 5*Ang]);
xlabel('Time (s)');
legend('Gait Phase', 'KAMoment', 'CAMoment', 'Fz', 'Lock Position',
'Pressure', 'State1', 'State2', 'State3', 'State4', 'State5', 'State6',
'State7', 'State8', 'StateU', 'AccX', 'AccZ', 'Ang');
axis([-inf inf -100 150]);
FileNameFig=asplssFile;
LengthFilename=length(asplssFile)-3;
FileNameFig = string(FileNameFig(1:LengthFilename));
FileNameFig=FileNameFig+'fig';
FileNameFig=char(FileNameFig);
saveas(h12, 'allsignals-partX-trialX.jpg');

%% Determine when KAMoment Logic == Ang Logic
% when plotting these (and the 2 plots below), plot only for stance phase
% FzLogic is used to plot only for stance phase
ang_correct_ext=((KAMomentLogicExt+AngLogicExt+FzLogic)==3);
ang_correct_flex=((KAMomentLogicFlex+AngLogicFlex+FzLogic)==3);
ang_incorrect_ext=((KAMomentLogicFlex+AngLogicExt+FzLogic)==3);
ang_incorrect_flex=((KAMomentLogicExt+AngLogicFlex+FzLogic)==3);
%% plot only KAMoment, Ang
h12=figure;
plot(xRange,[FzLogic*400-200,KAMoment,5*Ang, ang_correct_ext*3-50, ang_correct_flex*3-60,ang_incorrect_ext*3-70,ang_incorrect_flex*3-80 ]);xlabel('Time (s)'); legend('Gait event', 'KAMoment', 'Ang', 'ext', 'flex', 'inc-ext', 'inc-flex');axis([-inf inf -100 150]);saveas(h12,'ang-partX-trialX.jpg');

%% determining Ang threshold - delete later
figure
plot(xRange,Ang);
figure
plot(xRange, Pres);

%% Determine when KAMoment Logic == Pres Logic

% Determine when KAMoment Logic == Pres Logic

% Determine when KAMoment Logic == Pres Logic

%% plot only KAMoment, FSR
h13=figure;
hold on
plot(xRange,[KAMoment, Pres/50, KAMomentLogicFlex*400-200, PresLogicFlex*400-200 ]);plot(xRange,[FzLogic*400-200,KAMoment, Pres/50, correct_ext*3-20, correct_flex*3-30,incorrect_ext*3-40,incorrect_flex*3-50 ]);plot(xRange,KAFrictionMoment*OneArray); % plot KAMoment threshold plot(xRange,1*OneArray); % plot pressure sensor thresholdxlabel('Time (s)'); legend('Gait event', 'KAMoment', 'Pres', 'ext', 'flex', 'inc-ext', 'inc-flex');axis([-inf inf -100 100]);
hold off
saveas(h13,'pres-partX-trialX.jpg');

%% plot and compare FSR and Ang

% plot and compare FSR and Ang

% plot and compare FSR and Ang
legend('Gait event', 'Pres', 'Ang', 'pres-ext', 'ang-ext', 'pres-flex', 'ang-flex');
axis([-inf inf -100 100]);

hold off

% Write for FSR (pres) and Ang
filename_pres = 'ang-pres-comparison.csv'; % change this for each new participant
csvwrite(filename_pres, transpose(pres_ext), 1, 0);
dlmwrite(filename_pres, transpose(ang_ext), 'append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename_pres, transpose(pres_flex), 'append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename_pres, transpose(ang_flex), 'append', 'delimiter', ',', 'roffset', 0);

%% Determine when CAMoment Logic == Lock Logic
lock_correct_ext = ((CAMomentLogicExt + ProxLogicUnlock + FzLogic) == 3);
lock_correct_flex = ((CAMomentLogicFlex + ProxLogicLock + FzLogic) == 3);
lock_incorrect_ext = ((CAMomentLogicExt + ProxLogicLock + FzLogic) == 3);
lock_incorrect_flex = ((CAMomentLogicFlex + ProxLogicUnlock + FzLogic) == 3);

%% plot only CAMoment, prox (lock position)
h14 = figure;
hold on
plot(xRange, [CAMoment, (FzLogic*400)-200, Prox*10, lock_correct_ext*3-20, lock_correct_flex*3-30, lock_incorrect_ext*3-40, lock_incorrect_flex*3-50 ]); plot(xRange, CAFrictionMoment*OneArray); % plot CAMoment threshold plot(xRange, (Locked*10*OneArray));
xlabel('Time (s)');
legend('CAMoment', 'Gait event', 'Prox', 'ext', 'flex', 'inc-ext', 'inc-flex');
axis([-inf inf -100 100]);
hold off
saveas(h14, 'camoment-partX-trialX.jpg');

%% Write to file how well FSR and Ang predict ext/flex
% Write for FSR (pres)
filename_pres = 'participant5-pres.csv'; % change this for each new participant
csvwrite(filename_pres, transpose(correct_ext), 1, 0);
dlmwrite(filename_pres, transpose(correct_flex), 'append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename_pres, transpose(incorrect_ext), 'append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename_pres, transpose(incorrect_flex), 'append', 'delimiter', ',', 'roffset', 0);

%Write to same file for Ang
dlmwrite(filename_pres, transpose(ang_correct_ext), 'append', 'delimiter', ',', 'roffset', 5);
dlmwrite(filename_pres, transpose(ang_correct_flex), 'append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename_pres,transpose(ang_incorrect_ext),'-append',
'delimiter','','roffset',0);
dlmwrite(filename_pres,transpose(ang_incorrect_flex),'-append',
'delimiter','','roffset',0);

% Write to same file for Prox
dlmwrite(filename_pres,transpose(lock_correct_ext),'-append',
'delimiter','','roffset',5);
dlmwrite(filename_pres,transpose(lock_correct_flex),'-append',
'delimiter','','roffset',0);
dlmwrite(filename_pres,transpose(lock_incorrect_ext),'-append',
'delimiter','','roffset',0);
dlmwrite(filename_pres,transpose(lock_incorrect_flex),'-append',
'delimiter','','roffset',0);

end
Appendix C: MATLAB Code for State Analysis

```matlab
function [Counter] = stateanalysis(Counter,WriteData, StepInputs) %call function by' [Counter]=sensorState(Counter, WriteData, StepInputs)'
close all
%Counter is used for writing to lines and sheets in excell - set to 1
%Write data: 1 - Writes full data to Excel - SLOW; 0 - Only writes summary
%Data - FAST PROCESSING
%StepInputs: 1 - Prompts, 0 - takes data from Excell file cells J1 and K1
%where it has been
%Previously manually inputted. If no data was inputted, set StepInput to 1

% MODEL INPUTS set these values based on model % makes adjustment for lock
% spring based on lock position and hence lock spring contribution (i.e. 3Nm
% when fully unlocked) and friction measured to be 4Nm @ full knee extension
KASpringMoment = 1;% (1 Nm for ext assist, a
KASpringMoment = KASpringMoment - 0;% (1Nm, 2kg x 50mm for sencondary ext
spring
CASpringMoment = 2;% was measured at 2Nm
KAFrictionMoment = 0;% was measured at 2Nm
CAFrictionMoment = 0;% was measured at 0.5Nm
Unlocked = 1.5;% position of unlocked lock (1 mm typical)
Locked = 1.5;% position of locked (1mm to 0.5 when more securely locked)

% Processes and plots IPS, FSR, Accelerometer, and load transducer data
% Synchronizes ASPL-SS data with ATI data based on slope change within first
% 500 readings for AccX and FZ which correspond to stomp action
% Load all system data

[atiFile,atiPath] = uigetfile('*.csv','Select the ATI data file.');
[asplssFile,asplssPath] = uigetfile('*.csv','Select the ASPLSS data file.');

%[atiFile,atiPath] = uigetfile('*.csv','Select the ATI data file.');
atiData = csvread(fullfile(atiPath,atiFile));
% Load ASPLSS .csv file
%[asplssFile,asplssPath] = uigetfile('*.csv','Select the ASPLSS data file.');
asplssData = csvread(fullfile(asplssPath,asplssFile));

% Time stamps
% Calculate seconds since epoch day (00:00:00, 1st January 1990) to date
% and time for ATI data based on file name and Excel timestamps
dateA= regexp(atiFile,'(?<=\([^\)]*\)+(?=\))','match'); % regexp extracts
% date and time string from ATI filename
%Process date and time string to a format readable by MATLAB
dateA{1}(1:3)=''; % replaces day (columns 1-3) with ''
dateB=strrep(dateA{1},',EDT','');
dateC=strrep(dateB,':','');
ref=datevec('01-01-1900, 00:00:00'); % Transform epoch day into date vector
curr=datevec(dateC); % Transform current time into date vector
```
NTPTime=etime(cur,ref);  %Elapsed secs since epoch day/time to current
day/time
cur2=cur;
cur2(1,4:6)=0;
NTPTime2=etime(cur2,ref);  %Elapsed secs since epoch day/time to current day

% ATI
ModNTPTime = floor(NTPTime/(2^20))*(2^20);  %Masks lower 20 bits of NTP Time
CorNTPTime = ModNTPTime-10800;  %Corrects for EDT Timezone
ROTTime = atiData(:,1)./4096;  %Seconds since last rollover
ROTTime(ROTTime<0) = ROTTime(ROTTime<0) + 2^20;  % Make -ve timestamps +ve
atiTime = ROTTime+CorNTPTime;  %col vec = # secs since epoch time for each row
atiData(:,1) = atiTime;  %substitute col vec for original ATI time stamps

%% Synchronization

% ATI Fz
fzStartInd = 1;
atiTime = atiData(:,1)-atiData(fzStartInd,1);  %set sync timestamp = 0

%ASPLSS AccX
accZStartInd = 12;
asplassTime = asplssData(:,6)-double(asplssData(accZStartInd,6));  %set sync
timestamp = 0
asplassTime = asplssTime/1000.0;  %convert to seconds from ms

% ATI Fz
atiDataTrunc = atiData(1:1500,9);  %create a truncated matrix of ATI data
fzDiff = 0;
i = 1;
while (fzDiff) > -30000  %search for first instance of deltaFz > -30000
    i = i+1;
    fzDiff = atiDataTrunc(i)-atiDataTrunc(i-1);
end
fzStartInd = i-1;
atiTime = atiData(:,1)-atiData(fzStartInd,1);  %set sync timestamp = 0

%ASPLSS AccZ
asplassDataTrunc = asplssData(1:500,2);  %create a truncated matrix of ASPL-SS
data
accZDiff = 0;
j = 1;
while (accZDiff) < 40  %search for first instance of deltaAccZ > 40
    j = j+1;
    accZDiff = asplssDataTrunc(j)-asplssDataTrunc(j-1);
end
accZStartInd = j+3;
asplassTime = asplssData(:,6)-double(asplssData(accZStartInd,6));  %set sync
timestamp = 0
asplassTime = asplssTime/1000.0;  %convert to seconds from ms

%% neutralProx - Calibrates lock position measurement
neutralProx = max(asplssData(:,4));
%% neutralAng - Calibrates measurement from angled LPS
neutralAng = max(asplssData(:,5));

%% Data set variables
AccX = asplssData(:,2)/100; % Vertical acceleration
AccZ = asplssData(:,3)/100; % Horizontal acceleration
Prox = -((asplssData(:,4)-neutralProx)/80.55); % Lock position (mm)
Pres = asplssData(:,1); % FSR force
Ang = (asplssData(:,5)-neutralAng)/80.55; % angled sensor

% Cut off data before 0 timestamp
AccX=AccX(accZStartInd:length(asplssTime));
AccZ=AccZ(accZStartInd:length(asplssTime));
Prox=Prox(accZStartInd:length(asplssTime));
Pres=Pres(accZStartInd:length(asplssTime));
Ang=Ang(accZStartInd:length(asplssTime));

FX = atiData(:,7)/1000; % /1000 to correct for ATI errors
FY = atiData(:,8)/1000; % /1000 to correct for ATI errors
FZ = atiData(:,9)/1000; % /1000 to correct for ATI errors
TY = atiData(:,11)/1000; % /1000 to correct for ATI errors
DX = 0.0235;
DX2 = 0.02; % in flexion with lock in locked position
DZ = 0.06433;
DZ2 = 0.1953;
Theta = 0;

% Cut off data before 0 timestamp
FX=FX(fzStartInd:length(atiTime));
FY=FY(fzStartInd:length(atiTime));
FZ=FZ(fzStartInd:length(atiTime));
TY=TY(fzStartInd:length(atiTime));

% Synchronize start times to the 0 time point set
x1=asplssTime;
x2=atiTime;
minX1 = min(x1);
maxX1 = max(x1);
x1Range=minX1+maxX1;
x1=x1(accZStartInd:length(x1));

%x1 = x1-minX1;

minX2 = min(x2);
maxX2 = max(x2);
x2Range=minX2+maxX2;
%x2 = x2-minX2;
x2=x2(fzStartInd:length(x2));

figure
plot(x1,Prox);
hold on
plot(x1,AccX);
plot(x2,-FZ/100);
hold off

%% Interpolate values for Prox

x1Range= 0:0.01:x1Range;
x1Range=transpose(x1Range);
Prox=interp1(x1, Prox, x1Range, 'spline');
Ang=interp1(x1, Ang, x1Range, 'spline');
AccX=interp1(x1, AccX, x1Range, 'spline');
Pres=interp1(x1, Pres, x1Range, 'spline');
AccZ=interp1(x1, AccZ, x1Range, 'spline');

%% Interpolate values for ATI data

x2Range= 0:0.01:x2Range;
x2Range=transpose(x2Range);
FX=interp1(x2, FX, x2Range, 'spline');
FY=interp1(x2, FY, x2Range, 'spline');
FZ=interp1(x2, FZ, x2Range, 'spline');
TY=interp1(x2, TY, x2Range, 'spline');

%% Cut off ATI and ASPLSS data at same end point

% find minimum end time stamp between the two sensor systems
maxX= min(max(x2Range),max(x1Range));

% shorten xRanges to minimum end time between two systems
i=1;
while x1Range(i) ~= maxX
    i=i+1;
end

j=1;
% while x2Range(j) ~= maxX
%    j=j+1;
% end

x1Range=x1Range(1:i);
x2Range=x2Range(1:j);
xRange=x1Range;

% shorten data sets to minimum end time between two systems
Prox=Prox(1:i);
AccX=AccX(1:i);
AccZ=AccZ(1:i);
Ang=Ang(1:i);
Pres=Pres(1:i);

FX=FX(1:i);
FY=FY(1:i);
FZ=FZ(1:i);
Fz=FZ(1:i);
TY=TY(1:i);

figure
plot(xRange,Prox);
hold on
plot(xRange,AccX);

plot(xRange,-FZ/100);
hold off

%% ATI data origin adjustment

DX2=DX2+(Prox*2)/1000; % compensate for change in location of KA from loadcell due to movement at CA
DX2length= length(DX2);
CAMoment = (TY*cos(Theta))+(FX*cos(Theta))*DZ-(FY*sin(Theta))*DZ+FZ*DX;
% control axis
KAMoment = (TY*cos(Theta))+(FX*cos(Theta))*DZ2-(FY*sin(Theta))*DZ2+FZ.*DX2; % knee axis

%% Remove non steps prior to first step
figure
[h1]=plot(xRange,Fz,xRange,(Fz/100)+20);
axis([0 inf 0 100])
grid on;

if StepInputs==1

  % User-entered StartTime data start (s)
  prompt = {'Start time of steps'};
  dlg_title = 'Start time';
  num_lines = 1;
  def = {'4'};
  answer = inputdlg(prompt,dlg_title,num_lines,def);
  StartTime = str2double(answer{1})*100;

  % Set Fz value to be smaller (since Fz is negative) than the smallest value occurring during swing-phase
  % Fz is negative) swing-phase values
  prompt = {'Fz threshold'};
  dlg_title = 'Fz threshold';
  num_lines = 1;
  def = {'30'};
  answer = inputdlg(prompt,dlg_title,num_lines,def);
  FzThreshold = str2double(answer{1});

elseif StepInputs==0;
  StartTime = asplssData(1,6)*100;
  FzThreshold = asplssData(1,11);
end

% determine how many numbers are in xRange
LengthxRange=length(xRange);
% change xRange to go from start time, to end of range
xRange=xRange(StartTime:LengthxRange);
% change variables to show from start time to end of range
Prox=Prox(StartTime:LengthxRange);
Pres=Pres(StartTime:LengthxRange);
Ang=Ang(StartTime:LengthxRange);
KAMoment=KAMoment(StartTime:LengthxRange);
CAMoment=CAMoment(StartTime:LengthxRange);
DiffAccX=diff(AccX(StartTime-1:LengthxRange));
\[
\text{DiffAccZ} = \text{diff(AccZ(StartTime-1:LengthxRange))};
\]
\[
\text{AccX} = \text{AccX(StartTime:LengthxRange)};
\]
\[
\text{AccZ} = \text{AccZ(StartTime:LengthxRange)};
\]
\[
\text{DiffFz} = \text{Fz(StartTime-1:LengthxRange)};
\]
\[
\text{Fz} = \text{Fz(StartTime:LengthxRange)};
\]
\[
\text{DiffCAMomentLogic} = \text{diff(CAMoment)};
\]
\[
\text{DiffCAMomentLogicFlex} = -1*(\text{DiffCAMomentLogic} > 0);
\]
\[
\text{DiffCAMomentLogicExt} = (\text{DiffCAMomentLogic} < 0);
\]
\[
\text{DiffCAMomentLogic} = \text{DiffCAMomentLogicFlex} + \text{DiffCAMomentLogicExt};
\]
\[
\text{DiffKAMomentLogic} = \text{diff(KAMoment)};
\]
\[
\text{DiffKAMomentLogicFlex} = -1*(\text{DiffKAMomentLogic} > 0);
\]
\[
\text{DiffKAMomentLogicExt} = (\text{DiffKAMomentLogic} < 0);
\]
\[
\text{DiffKAMomentLogic} = \text{DiffKAMomentLogicFlex} + \text{DiffKAMomentLogicExt};
\]
\[
\% \text{ Extract exact steady state steps}
\]
\[
\text{DiffFzLogic} = \text{DiffFz<FzThreshold};
\]
\[
\text{DiffFz} = \text{diff(DiffFzLogic)};
\]
\[
\text{EventsLogic} = \text{DiffFz==1};
\]
\[
\text{figure};
\]
\[
\text{plotyy(xRange,DiffFz, xRange, Fz)};
\]
\[
\text{Events}=\text{xRange(EventsLogic)};
\]
\[
\text{StartEvents}=\text{int16(100*Events(1)-StartTime)};
\]
\[
\text{EndStep1}=\text{int16(100*Events(2)-StartTime)};
\]
\[
\text{EndStep2}=\text{int16(100*Events(3)-StartTime)};
\]
\[
\text{EndEvents}=\text{int16(100*Events(4)-StartTime)};
\]
\[
\text{Step1}=100*(\text{Events(2)-Events(1))};
\]
\[
\text{StepTime} = \text{single(EndEvents-StartEvents)/300};
\]
\[
\% \text{Extraction of 3 steps}
\]
\[
\text{xRange3Step} = \text{xRange(StartEvents:EndEvents)};
\]
\[
\text{Prox3Step} = \text{Prox(StartEvents:EndEvents)};
\]
\[
\text{Pres3Step} = \text{Pres(StartEvents:EndEvents)};
\]
\[
\text{Ang3Step} = \text{Ang(StartEvents:EndEvents)};
\]
\[
\text{DiffKAMomentLogic3Step} = \text{DiffKAMomentLogic(StartEvents:EndEvents)};
\]
\[
\text{KAMoment3Step} = \text{KAMoment(StartEvents:EndEvents)} - \text{KASpringMoment} + \text{KAFrictionMoment*DiffKAMomentLogic3Step};
\]
\[
\text{DiffCAMomentLogic3Step} = \text{DiffCAMomentLogic(StartEvents:EndEvents)};
\]
\[
\text{CAMoment3Step} = \text{CAMoment(StartEvents:EndEvents)} - \text{CASpringMoment*(Prox3Step)} + \text{CAFrictionMoment*DiffCAMomentLogic3Step};
\]
\[
\text{Accx3Step} = \text{Accx(StartEvents:EndEvents)};
\]
\[
\text{Accz3Step} = \text{Accz(StartEvents:EndEvents)};
\]
\[
\text{Fz3Step} = \text{Fz(StartEvents:EndEvents)};
\]
\[
\text{DiffAccx3Step} = \text{DiffAccx(StartEvents:EndEvents)};
\]
\[
\text{DiffAccz3Step} = \text{DiffAccz(StartEvents:EndEvents)};
\]
\[
\text{DiffFz3Step} = \text{Fz(StartEvents:EndEvents)};
\]
\[
\% \text{Identifying gait events with ACCx and ACCz}
\]
\[
\text{AccXEventVert} = \text{AccX > 1.5};
\]
\[
\text{AccZEventHor} = \text{AccZ > 1};
\]
\[
\% \text{Finite state analysis.}
\]
\[
\text{KAMomentLogicExt} = \text{KAMoment3Step > KAFrictionMoment};
\]
\[
\text{KAMomentLogicFlex} = \text{KAMoment3Step} < \text{KAFrictionMoment};
\]
\[
\text{CAMomentLogicExt} = \text{CAMoment3Step > CAFrictionMoment};
\]
\[
\text{CAMomentLogicFlex} = \text{CAMoment3Step} < \text{CAFrictionMoment};
\]
ProxLogicUnlock=Prox3Step>Unlocked;
ProxLogicLock=Prox3Step<Locked;%was 0.5
FzLogic=Fz3Step<FzThreshold;
PresLogicExt=Pres3Step>400;
PresLogicFlex=Pres3Step<400;
AngLogicExt=Ang3Step>-0.2;
AngLogicFlex=Ang3Step<-0.2;
OneArray=ones(length(xRange3Step),1);

% states defined by WGAT
WState1=((ProxLogicLock+PresLogicExt+FzLogic)==3);
WState2=((ProxLogicLock+PresLogicFlex+FzLogic)==3);
WState3=((ProxLogicUnlock+PresLogicExt+FzLogic)==3);
WState4=((ProxLogicUnlock+PresLogicFlex+FzLogic)==3);

% states by Andrysek et al
State1=((KAMomentLogicExt+CAMomentLogicExt+ProxLogicLock+FzLogic)==4);
State2=((KAMomentLogicExt+CAMomentLogicExt+ProxLogicUnlock+FzLogic)==4);
State3=((KAMomentLogicExt+CAMomentLogicFlex+ProxLogicLock+FzLogic)==4);
State4=((KAMomentLogicExt+CAMomentLogicFlex+ProxLogicUnlock+FzLogic)==4);
State5=((KAMomentLogicFlex+CAMomentLogicExt+ProxLogicLock+FzLogic)==4);
State6=((KAMomentLogicFlex+CAMomentLogicExt+ProxLogicUnlock+FzLogic)==4);
State7=((KAMomentLogicFlex+CAMomentLogicFlex+ProxLogicLock+FzLogic)==4);
State8=((KAMomentLogicFlex+CAMomentLogicFlex+ProxLogicUnlock+FzLogic)==4);
StateUnd = State1 + State2 + State3 + State4 + State5 + State6 + State7 + State8;
StateUnd = ~StateUnd.*FzLogic;
StateUnd=~StateUnd==0;

%State proportion of Stance-phase
StanceLength = sum(FzLogic);
State1OnStance = sum(State1)/StanceLength;
State2OnStance = sum(State2)/StanceLength;
State3OnStance = sum(State3)/StanceLength;
State4OnStance = sum(State4)/StanceLength;
State5OnStance = sum(State5)/StanceLength;
State6OnStance = sum(State6)/StanceLength;
State7OnStance = sum(State7)/StanceLength;
State8OnStance = sum(State8)/StanceLength;
StateUStance = sum(StateUnd)/StanceLength;

TotalStatesOn = State1OnStance + State2OnStance + State3OnStance + State4OnStance + State5OnStance + State6OnStance + State7OnStance + State8OnStance;
TotalStatesOn2367 = State2OnStance + State3OnStance + State6OnStance + State7OnStance;
TotalStatesOn234678 = State2OnStance + State3OnStance + State4OnStance + State6OnStance + State7OnStance + State8OnStance;

TotalStates=[State1OnStance,State2OnStance,State3OnStance,State4OnStance,State5OnStance,State6OnStance,State7OnStance,State8OnStance, StateUStance, TotalStatesOn, TotalStatesOn2367, TotalStatesOn234678]

%For plotting the states
State1Plot=State1*3+OneArray*-50;
State2Plot=State2*3+OneArray*-55;
State3Plot = State3*3 + OneArray* -60;
State4Plot = State4*3 + OneArray* -65;
State5Plot = State5*3 + OneArray* -70;
State6Plot = State6*3 + OneArray* -75;
State7Plot = State7*3 + OneArray* -80;
State8Plot = State8*3 + OneArray* -85;
StateUndPlot = StateUnd*3 + OneArray* -90;
WState1Plot = WState1*3 + OneArray* -95;
WState2Plot = WState2*3 + OneArray* -100;
WState3Plot = WState3*3 + OneArray* -105;
WState4Plot = WState4*3 + OneArray* -110;

StartX = min(xRange3Step);
xRange3StepZero = xRange3Step - StartX;

h12 = figure;
plot(xRange3StepZero, [FzLogic*400 - 200, KAMoment3Step, CAMoment3Step, -Fz3Step/10, Prox3Step*10, Pres3Step/50, State1Plot, State2Plot, State3Plot, State4Plot, State5Plot, State6Plot, State7Plot, State8Plot, StateUndPlot, 10*Acc23Step+90, WState1Plot, WState2Plot, WState3Plot, WState4Plot]);
xlabel('Time (s)');
legend('Gait Phase', 'KAMoment', 'CAMoment', 'Fz', 'Lock Position', 'Pressure', 'State1', 'State2', 'State3', 'State4', 'State5', 'State6', 'State7', 'State8', 'StateU', 'Acc2', 'WState1', 'WState2', 'WState3', 'WState4');
axis([126 inf inf -120 150]);
FileNameFig = asplssFile;
LengthFilename = length(asplssFile) - 3;
FileNameFig = string(FileNameFig(1:LengthFilename));
FileNameFig = char(FileNameFig);
saveas(h12, FileNameFig);

h13 = figure;
plot(xRange3StepZero, [FzLogic*400 - 200, State1Plot+100, State2Plot+100, State3Plot+100, State4Plot+100, State5Plot+100, State6Plot+100, State7Plot+100, State8Plot+100, WState1Plot+100, WState2Plot+100, WState3Plot+100, WState4Plot+100]);
xlabel('Time (s)');
legend('Gait Phase', 'State1', 'State2', 'State3', 'State4', 'State5', 'State6', 'State7', 'State8', 'WState1', 'WState2', 'WState3', 'WState4');
axis([-inf inf -20 60]);
saveas(h13, 'states.jpg');

%% Extract state data as percent of 100% gait cycle for each step

Start3Step1 = 1;
Start3Step2 = EndStep1 - StartEvents;
Start3Step3 = EndStep2 - StartEvents;
Start3Step4 = EndEvents - StartEvents;
NormalRange = 1:1:100; NormalRange = transpose(NormalRange); NormalRange = single(NormalRange);
Normlength = length(NormalRange);

%Step1
FzStep1 = Fz3Step(Start3Step1:Start3Step2);
FzLogicStep1 = FzStep1 < FzThreshold;
%Step1
State1Step1=State1(Start3Step1:Start3Step2);
State1Step1=State1Step1(FzLogicStep1);
State1Step1=double(State1Step1);
State1Step1=resample(State1Step1, 100, length(xRangeStep1));
State1Step1=int16(State1Step1);
State2Step1=State2(Start3Step1:Start3Step2);
State2Step1=State2Step1(FzLogicStep1);
State2Step1=double(State2Step1);
State2Step1=resample(State2Step1, 100, length(xRangeStep1));
State2Step1=int16(State2Step1);
State3Step1=State3(Start3Step1:Start3Step2);
State3Step1=State3Step1(FzLogicStep1);
State3Step1=double(State3Step1);
State3Step1=resample(State3Step1, 100, length(xRangeStep1));
State3Step1=int16(State3Step1);
State4Step1=State4(Start3Step1:Start3Step2);
State4Step1=State4Step1(FzLogicStep1);
State4Step1=double(State4Step1);
State4Step1=resample(State4Step1, 100, length(xRangeStep1));
State4Step1=int16(State4Step1);
State5Step1=State5(Start3Step1:Start3Step2);
State5Step1=State5Step1(FzLogicStep1);
State5Step1=double(State5Step1);
State5Step1=resample(State5Step1, 100, length(xRangeStep1));
State5Step1=int16(State5Step1);
State6Step1=State6(Start3Step1:Start3Step2);
State6Step1=State6Step1(FzLogicStep1);
State6Step1=double(State6Step1);
State6Step1=resample(State6Step1, 100, length(xRangeStep1));
State6Step1=int16(State6Step1);
State7Step1=State7(Start3Step1:Start3Step2);
State7Step1=State7Step1(FzLogicStep1);
State7Step1=double(State7Step1);
State7Step1=resample(State7Step1, 100, length(xRangeStep1));
State7Step1=int16(State7Step1);
State8Step1=State8(Start3Step1:Start3Step2);
State8Step1=State8Step1(FzLogicStep1);
State8Step1=double(State8Step1);
State8Step1=resample(State8Step1, 100, length(xRangeStep1));
State8Step1=int16(State8Step1);
WState1Step1=WState1(Start3Step1:Start3Step2);
WState1Step1=WState1Step1(FzLogicStep1);
WState1Step1=double(WState1Step1);
WState1Step1=resample(WState1Step1, 100, length(xRangeStep1));
WState1Step1=int16(WState1Step1);
WState2Step1=WState2(Start3Step1:Start3Step2);
WState2Step1=WState2Step1(FzLogicStep1);
WState2Step1=double(WState2Step1);
WState2Step1=resample(WState2Step1, 100, length(xRangeStep1));
WState2Step1=int16(WState2Step1);
WState3Step1=WState3(Start3Step1:Start3Step2);
WState3Step1=WState3Step1(FzLogicStep1);
WState3Step1=double(WState3Step1);
WState3Step1=resample(WState3Step1, 100, length(xRangeStep1));
WState3Step1=int16(WState3Step1);
WState4Step1=WState4(Start3Step1:Start3Step2);
WState4Step1=WState4Step1(FzLogicStep1);
WState4Step1=double(WState4Step1);
WState4Step1=resample(WState4Step1, 100, length(xRangeStep1));
WState4Step1=int16(WState4Step1);

%xStep2
FzStep2=Fz3Step(Start3Step2:Start3Step3);
FzLogicStep2=FzStep2<FzThreshold;
xRangeStep2=xRange3Step(Start3Step2:Start3Step3);
xRangeStep2Length=length(xRangeStep2);
xRangeStep2=xRangeStep2(FzLogicStep2);
xRangeStance2Length=length(xRangeStep2);
xRangeStep2=(1:1:xRangeStance2Length);
PercentStanceStep2=xRangeStance2Length/xRangeStep2Length;

State1Step2=State1(Start3Step2:Start3Step3);
State1Step2=State1Step2(FzLogicStep2);
State1Step2=double(State1Step2);State1Step2=resample(State1Step2, 100, length(xRangeStep2)); State1Step2=int16(State1Step2);

State2Step2=State2(Start3Step2:Start3Step3);
State2Step2=State2Step2(FzLogicStep2);
State2Step2=double(State2Step2);State2Step2=resample(State2Step2, 100, length(xRangeStep2)); State2Step2=int16(State2Step2);

State3Step2=State3(Start3Step2:Start3Step3);
State3Step2=State3Step2(FzLogicStep2);
State3Step2=double(State3Step2);State3Step2=resample(State3Step2, 100, length(xRangeStep2)); State3Step2=int16(State3Step2);

State4Step2=State4(Start3Step2:Start3Step3);
State4Step2=State4Step2(FzLogicStep2);
State4Step2=double(State4Step2);State4Step2=resample(State4Step2, 100, length(xRangeStep2)); State4Step2=int16(State4Step2);

State5Step2=State5(Start3Step2:Start3Step3);
State5Step2=State5Step2(FzLogicStep2);
State5Step2=double(State5Step2);State5Step2=resample(State5Step2, 100, length(xRangeStep2)); State5Step2=int16(State5Step2);

State6Step2=State6(Start3Step2:Start3Step3);
State6Step2=State6Step2(FzLogicStep2);
State6Step2=double(State6Step2);State6Step2=resample(State6Step2, 100, length(xRangeStep2)); State6Step2=int16(State6Step2);

State7Step2=State7(Start3Step2:Start3Step3);
State7Step2=State7Step2(FzLogicStep2);
State7Step2=double(State7Step2);State7Step2=resample(State7Step2, 100, length(xRangeStep2)); State7Step2=int16(State7Step2);

State8Step2=State8(Start3Step2:Start3Step3);
State8Step2=State8Step2(FzLogicStep2);
State8Step2=double(State8Step2);State8Step2=resample(State8Step2, 100, length(xRangeStep2)); State8Step2=int16(State8Step2);

WState1Step2=WState1(Start3Step2:Start3Step3);
WState1Step2=WState1Step2(FzLogicStep2);
WState1Step2=double(WState1Step2);WState1Step2=resample(WState1Step2, 100, length(xRangeStep2)); WState1Step2=int16(WState1Step2);

WState2Step2=WState2(Start3Step2:Start3Step3);
WState2Step2=WState2Step2(FzLogicStep2);
WState2Step2=double(WState2Step2);WState2Step2=resample(WState2Step2, 100, length(xRangeStep2)); WState2Step2=int16(WState2Step2);

WState3Step2=WState3(Start3Step2:Start3Step3);
WState3Step2=WState3Step2(FzLogicStep2);
WState3Step2=double(WState3Step2);WState3Step2=resample(WState3Step2, 100, length(xRangeStep2)); WState3Step2=int16(WState3Step2);

WState4Step2=WState4(Start3Step2:Start3Step3);
WState4Step2=WState4Step2(FzLogicStep2);
WState4Step2=double(WState4Step2);WState4Step2=resample(WState4Step2, 100, length(xRangeStep2)); WState4Step2=int16(WState4Step2);

%Step3
FzStep3=Fz3Step(Start3Step3:Start3Step4);
FzLogicStep3=FzStep3<FzThreshold;
xRangeStep3=xRange3Step3(Start3Step3:Start3Step4);
xRangeStep3Length=length(xRangeStep3);
xRangeStep3=xRangeStep3(FzLogicStep3);
xRangeStance3Length=length(xRangeStep3);
xRangeStep3=(1:1:xRangeStance3Length);
PercentStanceStep3=xRangeStance3Length/xRangeStep3Length;

State1Step3=State1(Start3Step3:Start3Step4);
State1Step3=State1Step3(FzLogicStep3);
State1Step3=double(State1Step3);
State1Step3=resample(State1Step3, 100,
length(xRangeStep3));
State1Step3=int16(State1Step3);
State2Step3=State2(Start3Step3:Start3Step4);
State2Step3=State2Step3(FzLogicStep3);
State2Step3=double(State2Step3);
State2Step3=resample(State2Step3, 100,
length(xRangeStep3));
State2Step3=int16(State2Step3);
State3Step3=State3(Start3Step3:Start3Step4);
State3Step3=State3Step3(FzLogicStep3);
State3Step3=double(State3Step3);
State3Step3=resample(State3Step3, 100,
length(xRangeStep3));
State3Step3=int16(State3Step3);
State4Step3=State4(Start3Step3:Start3Step4);
State4Step3=State4Step3(FzLogicStep3);
State4Step3=double(State4Step3);
State4Step3=resample(State4Step3, 100,
length(xRangeStep3));
State4Step3=int16(State4Step3);
State5Step3=State5(Start3Step3:Start3Step4);
State5Step3=State5Step3(FzLogicStep3);
State5Step3=double(State5Step3);
State5Step3=resample(State5Step3, 100,
length(xRangeStep3));
State5Step3=int16(State5Step3);
State6Step3=State6(Start3Step3:Start3Step4);
State6Step3=State6Step3(FzLogicStep3);
State6Step3=double(State6Step3);
State6Step3=resample(State6Step3, 100,
length(xRangeStep3));
State6Step3=int16(State6Step3);
State7Step3=State7(Start3Step3:Start3Step4);
State7Step3=State7Step3(FzLogicStep3);
State7Step3=double(State7Step3);
State7Step3=resample(State7Step3, 100,
length(xRangeStep3));
State7Step3=int16(State7Step3);
State8Step3=State8(Start3Step3:Start3Step4);
State8Step3=State8Step3(FzLogicStep3);
State8Step3=double(State8Step3);
State8Step3=resample(State8Step3, 100,
length(xRangeStep3));
State8Step3=int16(State8Step3);
WState1Step3=WState1(Start3Step3:Start3Step4);
WState1Step3=WState1Step3(FzLogicStep3);
WState1Step3=double(WState1Step3);
WState1Step3=resample(WState1Step3, 100,
length(xRangeStep3));
WState1Step3=int16(WState1Step3);
WState2Step3=WState2(Start3Step3:Start3Step4);
WState2Step3=WState2Step3(FzLogicStep3);
WState2Step3=double(WState2Step3);
WState2Step3=resample(WState2Step3, 100,
length(xRangeStep3));
WState2Step3=int16(WState2Step3);
WState3Step3=WState3(Start3Step3:Start3Step4);
WState3Step3=WState3Step3(FzLogicStep3);
WState3Step3=double(WState3Step3);
WState3Step3=resample(WState3Step3, 100,
length(xRangeStep3));
WState3Step3=int16(WState3Step3);
WState4Step3=WState4(Start3Step3:Start3Step4);
WState4Step3=WState4Step3(FzLogicStep3);
WState4Step3=double(WState4Step3);
WState4Step3=resample(WState4Step3, 100,
length(xRangeStep3));
WState4Step3=int16(WState4Step3);

State13Steps=State1Step1+State1Step2+State1Step3;
State23Steps=State2Step1+State2Step2+State2Step3;
State3Steps = State3Step1 + State3Step2 + State3Step3;
State4Steps = State4Step1 + State4Step2 + State4Step3;
State5Steps = State5Step1 + State5Step2 + State5Step3;
State6Steps = State6Step1 + State6Step2 + State6Step3;
State7Steps = State7Step1 + State7Step2 + State7Step3;
State8Steps = State8Step1 + State8Step2 + State8Step3;
WState1Steps = WState1Step1 + WState1Step2 + WState1Step3;
WState2Steps = WState2Step1 + WState2Step2 + WState2Step3;
WState3Steps = WState3Step1 + WState3Step2 + WState3Step3;
WState4Steps = WState4Step1 + WState4Step2 + WState4Step3;

%% Excel Write

Counter = Counter + 1;
file = string(asplssFile);
filename = file + 'Result-Sensor.csv';
csvwrite(filename, TotalStates, 1, 0);
dlmwrite(filename, StepTime, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, PercentStanceStep1, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, PercentStanceStep2, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, PercentStanceStep3, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, KASpringMoment, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, CASpringMoment, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, KAFrictionMoment, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, CAFrictionMoment, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, Unlocked, '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, Locked, '-append', 'delimiter', ',', 'roffset', 0);
if WriteData == 1;

% write step data for states with 3 steps Added

filename = file + 'State-Results.csv';
csvwrite(filename, transpose(State1Steps), 1, 0);
dlmwrite(filename, transpose(State2Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State3Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State4Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State5Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State6Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State7Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(State8Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(WState1Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(WState2Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(WState3Steps), '-append', 'delimiter', ',', 'roffset', 0);
dlmwrite(filename, transpose(WState4Steps), '-append', 'delimiter', ',', 'roffset', 0);

% All data

% create new file for all data
filename2 = file + 'All-Data.csv';
% write data to new .csv file
% data is written in rows, each new row is a new data set
% note - there is no labels on the data. Add manually post-processing
csvwrite(filename2,transpose(xRange),1,0);
dlmwrite(filename2,transpose(xRange3StepZero), '-append','delimiter','

end
Appendix D: Design of User Interface

This project was completed by Chaojin (Jessica) Jia, Vidhant Pal, Hayden Provias, and Rachel Reding (the author) for BME1802, Human Factors of Medical Devices. The specific need addressed in this project was the need to create an effective user interface (UI) that communicates the status of the wearable gait analysis tool to clinicians and researchers during gait analysis to ensure data is acquired for the purposes of improving walking difficulties for amputees.

The functions of the UI were to turn the device on/off, inform user that calibration is complete (assuming calibration begins automatically when powered on), turn recording on/off, and inform user of low battery. The objectives were to minimize the use of language on the UI, maximize intuitiveness (defined as ease of use), and minimize the size of the UI.

The team performed a combination of both empirical and analytical analyses to gain insight on the path to a viable solution. This included a functional decomposition, heuristic analysis, hierarchical task analysis, fault tree analysis, user interviews and formative evaluations (i.e., design critique). In conjunction with this analysis, several versions of the prototype were created, including a low fidelity, non-functioning prototype (see Figure 30) to gain insights about the overall layout and symbol selection, all the way to fully functional prototypes (see Figure 31) for the user testing. The functional prototypes used 3D-printed mounts that housed LEDs and buttons, controlled by an Arduino Uno. In developing the prototypes, the author contributed to the physical construction of the prototype (i.e., cutting and drawing the cardboard prototypes, and designing the 3D-printed prototypes in Solidworks). All team members presented the prototypes at the Demonstration Presentation, which resulted in further design suggestions that were incorporated into the final design, which the author modified in Solidworks (see Figure 32). Some of the major features of the final design were an LED bar graph to display discrete battery levels with red bars indicating low battery, extruded symbols of a different colour than the mount colour for ease of viewing, and a non-latching button for recording in case users turn the device off before stopping recording, eliminating confusion if they turn the device on again and recording starts automatically.
**Figure 30:** Three low-fidelity prototypes used for the design critique and cognitive walkthrough. Some of the distinct features were: *(left)* using a linear layout, having a separate button for recording, and using a checkmark for calibration; *(middle)* using a linear layout, and using a “refresh” symbol for calibration; and *(right)* using a single Red-Green-Blue (RGB) LED to communicate all functions.

**Figure 31:** Two higher-fidelity prototypes. Some of the distinct features were: *(left)* symbols directly on buttons, and sequential symbols directly on the interface; *(right)* symbols directly on the interface, user actions and indicators separated by solid line.
When concluding the project, the team had several suggestions for future directions:

1. Since this prototype was the first instance of a user interface on the wearable gait analysis tool, more emphasis was placed on the development of a working prototype which could be tested and refined through formative evaluation as well as the development of the critical tasks associated with the device for future validation and verification testing of the device. However, for future testing on the device, comparisons between the designed user interface as well as interfaces of similar tools should be performed to further direct the development of the device.

2. Furthermore, in the selection of components for the UI, 3D-printable plastic, LEDs and buttons were chosen to maximize durability so that the device is usable in various climates and environments, on par with that of the AT-Knee. However, the durability of the device was not quantified or studied, and this is something to consider in future.

3. Moreover, there were significant issues obtaining consensus on a “calibrated” symbol, which remained a challenge for the team. Despite asking the critiquers at the Design Critique, surveying lay people, and questioning users during the Usability Testing, the team did not conclusively determine a symbol that was easily recognizable by the
majority of users. Therefore, future research is needed to determine the best representation of “calibrated”.

4. Lastly, a physical prototype should be created incorporating all the features of the final prototype. The UI should be integrated with the entire gait analysis tool and user testing should be completed with users performing tasks with the entire system. This could be an important comparison to determine the usability of the device with and without a user interface. While we obtained qualitative feedback in the form of feedback, and quantitative feedback by counting errors, a useful comparison might be the time to complete a task with and without the user interface. Further, a limitation of the current usability testing that was done is that it was a simulated scenario with the UI in isolation, rather than having users perform tasks with the entire system. Performing tasks with the entire system might uncover more usability issues that could be solved with design changes to the UI. Moreover, the current usability testing only had users in a post-graduate university degree in Canada, all who were fluent in English. As this is a device that might be sent to users who are in other countries and may not fluent in English, it is imperative to perform usability testing with participants who are from other countries and/or not fluent in English. This is especially important to ensure the meanings of the symbols remain clear to these users, especially the “o REC” symbol as it has some linguistic use.
Appendix E: Technical Documentation for Linear Position Sensor

**Specifications**

<table>
<thead>
<tr>
<th>Model</th>
<th>9605</th>
<th>9610</th>
<th>9615</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part Number</td>
<td>9605R1.7KL2.0</td>
<td>9610R3.4KL2.0</td>
<td>9615R5.1KL2.0</td>
</tr>
<tr>
<td>Total Electrical Travel (A) inches (mm)</td>
<td>0.50 (12.7)</td>
<td>1.00 (25.4)</td>
<td>1.50 (38.1)</td>
</tr>
<tr>
<td>Total DC Resistance ± 25%</td>
<td>1.7K</td>
<td>3.4K</td>
<td>5.7K</td>
</tr>
<tr>
<td>Linearity Over Active Electrical Travel</td>
<td>± 2%</td>
<td>± 2%</td>
<td>± 0.5%</td>
</tr>
<tr>
<td>Best Practical Linearity (Option)</td>
<td>± 1.5%</td>
<td>± 0.5%</td>
<td>± 0.35%</td>
</tr>
<tr>
<td>Power Rating at 77ºC, Watts</td>
<td>0.25</td>
<td>0.50</td>
<td>0.75</td>
</tr>
<tr>
<td>Mechanical Travel E (9160E ± 0.4) (B) inches (mm)</td>
<td>0.56 (14.2)</td>
<td>1.06 (26.9)</td>
<td>1.56 (39.6)</td>
</tr>
<tr>
<td>Housing Length E 9160 (E ± 0.4) (C) inches (mm)</td>
<td>1.06 (26.9)</td>
<td>1.56 (39.6)</td>
<td>2.06 (52.3)</td>
</tr>
<tr>
<td>Terminal Spacing (D) inches (mm)</td>
<td>0.30 (7.6)</td>
<td>0.50 (12.7)</td>
<td>0.80 (20.3)</td>
</tr>
<tr>
<td>(E) inches (mm)</td>
<td>0.20 (5.1)</td>
<td>0.50 (12.7)</td>
<td>0.70 (17.8)</td>
</tr>
<tr>
<td>Fully Extended Length 2.015 (± 0.4) (F) inches (mm)</td>
<td>0.810 (20.6)</td>
<td>1.310 (33.3)</td>
<td>1.810 (46.0)</td>
</tr>
<tr>
<td>Mechanical Life</td>
<td>1,000,000 Full Cycles, 5,000,000 Dither Cycles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop Strength oz. (Newtons)</td>
<td>360 (100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actuation Force oz. (Newtons)</td>
<td>14.4 (4.0) Maximum, supplied with internal spring to return actuator to extended position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Humidity</td>
<td>95% RH, 68ºF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vibration (G) g's</td>
<td>Up to ±50g, 5s max, 100-2000Hz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shock</td>
<td>Up to ±25g 2ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temperature Limits</td>
<td>-40ºC to +135ºC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The BEI Sensors 9600 Series Linear Position Sensor is ideally suited for use where reliability in a harsh operating environment is a primary consideration.

Industrial, vehicular, appliance, machine tool and robotic applications benefit from the unit’s high temperature stability. The solderable terminal tabs, suitable for use with .110” (2.8mm) PDQ styled crimped wiring lugs, and durable spring-loaded plunger, make installation and operation easy.

The 9600 Series is available in three standard sizes and provides excellent life at 1,000,000 full cycles (5 million dither cycles), which means high reliability in demanding installations.
Offering one of the smallest form factors available, the 9600 Series spring return linear motion position sensor is ideal for a variety of applications requiring a highly miniaturized solution. Reliably designed to deliver precision feedback, this innovative potentiometer offers a low-cost solution that is ideally suited for use with microprocessor-based systems, including joystick controls, robotics and industrial automation and controls.

An integral slider/contact assembly assures smooth, noise-free travel over the unit’s proprietary infinite resolution element to complement sensitive systems controls. Compatible with leading industry-standard terminations, the 9600 is designed to provide virtual plug-and-play installation simplicity. For abnormally tight packaging constraints, a new 0.15” (3.8mm) short length terminal tab configuration is available. Durably made of high-temperature stable materials, the 9600 Series offers a highly ruggedized design where reliability in a harsh environment is a primary consideration.

Ideal for volume use and replacement of other rotary or linear feedback devices, all standard 9600 models are available immediately from stock. Special electrical and mechanical performance characteristics and packaging configurations are available.

Benefits:

- Miniature size ideal for tight spaces
- Spring return design allows interface-free installation
- 0.50” to 1.50” electrical travel models for design versatility
- Accepts industry-standard flat terminals; ultra short terminal style available
- Long operating life 2 million cycles (5 million dither cycles)

ISO9001 Certified/QS9000 Compliant
## Specifications (Typical)*

<table>
<thead>
<tr>
<th>MODEL</th>
<th>9605</th>
<th>9610</th>
<th>9615</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Electrical Travel (inches)</td>
<td>0.50 (12.7)</td>
<td>1.00 (25.4)</td>
<td>1.50 (38.1)</td>
</tr>
<tr>
<td>Active Electrical Travel (Fig. 1)</td>
<td>0.40 (10.0)</td>
<td>0.90 (22.8)</td>
<td>1.40 (35.6)</td>
</tr>
<tr>
<td>Total DC Resistance ± 25%</td>
<td>1.7K</td>
<td>3.4K</td>
<td>5.1K</td>
</tr>
<tr>
<td>Linearity Over Active Electrical Travel</td>
<td>± 2%</td>
<td>± 2%</td>
<td>± 2%</td>
</tr>
<tr>
<td>Best Practical Linearity</td>
<td>± 1.0%</td>
<td>± 0.5%</td>
<td>± 0.35%</td>
</tr>
<tr>
<td>Power Rating At 70°C, Watts</td>
<td>0.25</td>
<td>0.50</td>
<td>0.75</td>
</tr>
<tr>
<td>Mechanical Travel ± 0.015 (± 0.4)</td>
<td>0.56 (14.2)</td>
<td>0.96 (24.9)</td>
<td>1.56 (39.6)</td>
</tr>
<tr>
<td>Housing Length ± 0.15 (± 0.4)</td>
<td>1.06 (26.9)</td>
<td>1.56 (39.6)</td>
<td>2.06 (52.3)</td>
</tr>
<tr>
<td>Terminal Spacing:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(D) inches (mm)</td>
<td>0.30 (7.6)</td>
<td>0.50 (12.7)</td>
<td>0.80 (20.3)</td>
</tr>
<tr>
<td>(P) inches (mm)</td>
<td>0.20 (5.1)</td>
<td>0.30 (7.6)</td>
<td>0.70 (17.8)</td>
</tr>
<tr>
<td>Fully Extended Length ± 0.15 (± 0.4)</td>
<td>0.810 (20.6)</td>
<td>1.310 (33.3)</td>
<td>1.810 (46.0)</td>
</tr>
<tr>
<td>Terminal Length (Q Inches) (mm)</td>
<td>280 (7.1)</td>
<td>280 (7.1)</td>
<td>280 (7.1)</td>
</tr>
<tr>
<td>Short &quot;S&quot; Option</td>
<td>150 (3.8)</td>
<td>150 (3.8)</td>
<td>150 (3.8)</td>
</tr>
<tr>
<td>Mechanical Life</td>
<td>2,000,000 Full Cycles, 5,000,000 Other Cycles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shock Strength (oz) (Newton)</td>
<td>360 (100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actuation Force (oz) (Newton)</td>
<td>14.4 (4.0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Humidity</td>
<td>95% @ 88°F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vibration</td>
<td>15G, 50 to 1,000Hz, 2hrs. each plane</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shock</td>
<td>Up to 150G</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temperature Limits</td>
<td>-40°C to 135°C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Most specifications may be altered to meet specific requirements.

### Tolerances unless otherwise specified: ± .03 (± 0.8)

## Ordering Information

- **Basic Model**
  - 05 - 0.5 total electrical travel
  - 15 - 1.5 total electrical travel

- **Total Resistance**
  - 9605 = 1.7
  - 9610 = 3.4
  - 9615 = 5.1

- **Independent Linearity**
  - ± 2% standard on all models

Optional Terminal Lengths (all models)
- Long = 25 (6.4) - Standard (leave blank)
- Short = 15 (3.8) - "S"

*Consult Factory for availability of special resistance and linearity requirements.
## Appendix F: Results from Calibration of LPS

**Table 21:** Average experimental results from the three trials of the distance test.

<table>
<thead>
<tr>
<th>Distance (mm)</th>
<th>Theoretical (bits)</th>
<th>Average Experimental (bits)</th>
<th>Standard Deviation (bits)</th>
<th>Negative Error (bits)</th>
<th>Positive Error (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.24</td>
<td>99.882</td>
<td>100</td>
<td>0</td>
<td>96.886</td>
<td>102.878</td>
</tr>
<tr>
<td>1.44</td>
<td>115.992</td>
<td>116.667</td>
<td>0.577</td>
<td>112.512</td>
<td>119.472</td>
</tr>
<tr>
<td>1.64</td>
<td>132.102</td>
<td>131</td>
<td>0</td>
<td>128.139</td>
<td>136.065</td>
</tr>
<tr>
<td>1.84</td>
<td>148.212</td>
<td>148.667</td>
<td>0.577</td>
<td>143.766</td>
<td>152.658</td>
</tr>
<tr>
<td>2.04</td>
<td>164.322</td>
<td>163.333</td>
<td>0.577</td>
<td>159.392</td>
<td>169.252</td>
</tr>
<tr>
<td>2.24</td>
<td>180.432</td>
<td>178</td>
<td>0</td>
<td>175.019</td>
<td>185.845</td>
</tr>
<tr>
<td>2.44</td>
<td>196.542</td>
<td>195</td>
<td>0</td>
<td>190.646</td>
<td>202.438</td>
</tr>
<tr>
<td>2.64</td>
<td>212.652</td>
<td>210.667</td>
<td>0.577</td>
<td>206.272</td>
<td>219.032</td>
</tr>
<tr>
<td>2.84</td>
<td>228.762</td>
<td>226.667</td>
<td>0.577</td>
<td>221.899</td>
<td>235.625</td>
</tr>
<tr>
<td>3.04</td>
<td>244.872</td>
<td>243.333</td>
<td>1.154</td>
<td>237.526</td>
<td>252.218</td>
</tr>
<tr>
<td>3.24</td>
<td>260.982</td>
<td>259</td>
<td>0</td>
<td>253.152</td>
<td>268.811</td>
</tr>
<tr>
<td>3.44</td>
<td>277.092</td>
<td>274</td>
<td>0</td>
<td>268.779</td>
<td>285.405</td>
</tr>
<tr>
<td>3.64</td>
<td>293.202</td>
<td>291.333</td>
<td>1.155</td>
<td>284.406</td>
<td>301.998</td>
</tr>
<tr>
<td>3.84</td>
<td>309.312</td>
<td>307</td>
<td>0</td>
<td>300.033</td>
<td>318.591</td>
</tr>
<tr>
<td>4.04</td>
<td>325.422</td>
<td>323</td>
<td>0</td>
<td>315.659</td>
<td>335.185</td>
</tr>
<tr>
<td>4.24</td>
<td>341.532</td>
<td>338.667</td>
<td>1.155</td>
<td>331.286</td>
<td>351.778</td>
</tr>
<tr>
<td>4.44</td>
<td>357.642</td>
<td>354.333</td>
<td>0.577</td>
<td>346.913</td>
<td>368.371</td>
</tr>
<tr>
<td>4.64</td>
<td>373.752</td>
<td>371.333</td>
<td>0.577</td>
<td>362.539</td>
<td>384.964</td>
</tr>
<tr>
<td>4.84</td>
<td>389.862</td>
<td>388</td>
<td>0</td>
<td>378.166</td>
<td>401.558</td>
</tr>
<tr>
<td>5.04</td>
<td>405.972</td>
<td>405.333</td>
<td>0.577</td>
<td>393.793</td>
<td>418.151</td>
</tr>
<tr>
<td>5.24</td>
<td>422.082</td>
<td>422</td>
<td>1</td>
<td>409.420</td>
<td>434.744</td>
</tr>
</tbody>
</table>
Appendix G: Percentage of Stance Phase for 8-State Model

Table 22: Percentage of stance phase for states 2, 3, 6, and 7, and 2, 3, 4, 6, 7 and 8 of the 8-state model.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Combination of States</th>
<th>2+3+6+7</th>
<th>2+3+4+6+7+8</th>
<th>2+3+6+7+8</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>0.924</td>
<td>1.000</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>AB2</td>
<td>0.857</td>
<td>0.983</td>
<td>0.970</td>
<td></td>
</tr>
<tr>
<td>AB3</td>
<td>0.917</td>
<td>0.967</td>
<td>0.947</td>
<td></td>
</tr>
<tr>
<td>AB4</td>
<td>0.870</td>
<td>0.923</td>
<td>0.923</td>
<td></td>
</tr>
<tr>
<td>AB5</td>
<td>0.827</td>
<td>0.997</td>
<td>0.993</td>
<td></td>
</tr>
<tr>
<td>AB6</td>
<td>0.923</td>
<td>0.973</td>
<td>0.966</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>0.886</td>
<td>0.974</td>
<td>0.967</td>
<td></td>
</tr>
<tr>
<td>SD</td>
<td>0.041</td>
<td>0.028</td>
<td>0.029</td>
<td></td>
</tr>
</tbody>
</table>
Appendix H: Analysis of Extension and Flexion of WGAT Compared to Load Cell

**Table 23:** Percentage of stance phase spent in extension and flexion as measured by the load transducer and the WGAT.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Load transducer</th>
<th>WGAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1AB</td>
<td>Ext 0.44</td>
<td>Ext 0.45</td>
</tr>
<tr>
<td></td>
<td>Flex 0.56</td>
<td>Flex 0.55</td>
</tr>
<tr>
<td>2AB</td>
<td>Ext 0.61</td>
<td>Ext 0.53</td>
</tr>
<tr>
<td></td>
<td>Flex 0.39</td>
<td>Flex 0.47</td>
</tr>
<tr>
<td>3AB</td>
<td>Ext 0.67</td>
<td>Ext 0.58</td>
</tr>
<tr>
<td></td>
<td>Flex 0.33</td>
<td>Flex 0.42</td>
</tr>
<tr>
<td>4AB</td>
<td>Ext 0.37</td>
<td>Ext 0.31</td>
</tr>
<tr>
<td></td>
<td>Flex 0.63</td>
<td>Flex 0.69</td>
</tr>
<tr>
<td>5AB</td>
<td>Ext 0.52</td>
<td>Ext 0.34</td>
</tr>
<tr>
<td></td>
<td>Flex 0.48</td>
<td>Flex 0.66</td>
</tr>
<tr>
<td>6AB</td>
<td>Ext 0.56</td>
<td>Ext 0.43</td>
</tr>
<tr>
<td></td>
<td>Flex 0.44</td>
<td>Flex 0.57</td>
</tr>
</tbody>
</table>

**Table 24:** Mean difference in percentage of gait cycle spent in flexion, as calculated from data collected by the load transducer subtracted from either the angled LPS or FSR. SD is the standard deviation. Positive values indicate over-detection of flexion by the sensor as compared to the load transducer.

<table>
<thead>
<tr>
<th>Participant</th>
<th>Sensor</th>
<th>Mean difference in detection of flexion (%)</th>
<th>SD (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1AB</td>
<td>Angled LPS</td>
<td>2.22</td>
<td>4.56</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>2.30</td>
<td>4.11</td>
</tr>
<tr>
<td>2AB</td>
<td>Angled LPS</td>
<td>14.14</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>7.56</td>
<td>4.76</td>
</tr>
<tr>
<td>3AB</td>
<td>Angled LPS</td>
<td>25.44</td>
<td>6.76</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>10.09</td>
<td>8.89</td>
</tr>
<tr>
<td>4AB</td>
<td>Angled LPS</td>
<td>11.27</td>
<td>3.80</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>5.81</td>
<td>2.67</td>
</tr>
<tr>
<td>5AB</td>
<td>Angled LPS</td>
<td>27.47</td>
<td>11.76</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>20.96</td>
<td>10.88</td>
</tr>
<tr>
<td>6AB</td>
<td>Angled LPS</td>
<td>4.50</td>
<td>11.91</td>
</tr>
<tr>
<td></td>
<td>FSR</td>
<td>8.92</td>
<td>7.54</td>
</tr>
</tbody>
</table>
Appendix I: Sensitivity Analysis

Table 25: Angled LPS sensitivity analysis. Maximum accuracy across all participants was for a threshold of -0.19 mm, yielding an accuracy of 78.2%.

![Graph showing mean accuracy vs threshold for Angled LPS sensitivity analysis.]

Table 26: FSR sensitivity analysis. Maximum accuracy across all participants was for a threshold of 380 bits, yielding an accuracy of 83.4%.

![Graph showing mean accuracy vs threshold for FSR sensitivity analysis.]