BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Thesis

ON THE DEVELOPMENT OF SELF-PACED TREADMILL SYSTEMS FOR GAIT RESEARCH

by

ZOE PERKINS

B.A., Boston University, 2021

Submitted in partial fulfillment of the

requirements for the degree of

Master of Science

2025

Approved by

First Reader

Cara L. Lewis, Ph.D., PT
Professor of Physical Therapy
Professor of Health Sciences
Associate Professor of Medicine

Second Reader

Sheryl Grace, Ph.D.
Associate Professor of Mechanical Engineering

Third Reader

Roberto Tron, Ph.D.
Associate Professor of Mechanical Engineering
Associate Professor of Systems Engineering

Acknowledgments

There are far more individuals than I will name here who have supported me throughout this research and my academic journey. To all the professors, advisors, staff, friends and family who have been a part of this journey - thank you.

I am especially grateful to my advisor and mentor, Dr. Cara Lewis. Over several years, she has generously shared her expertise in human movement and research, teaching me not what to think, but how. Her support and mentorship have been invaluable, and I am most grateful. I would also like to acknowledge my thesis committee - Drs. Sheryl Grace, Roberto Tron and Cara Lewis - for their guidance and feedback throughout the research and writing process, strengthening this work in meaningful ways. Of course, this work would not have been possible without Dr. Seungmoon Song, whose prior research formed the backbone of the updated SPT system implemented here. His generosity in sharing resources and expertise was critical, and I am appreciative.

I am also especially grateful to those who supported my day-to-day life throughout the completion of this thesis and my degree. To my parents - thank you for your love, encouragement and steadfast support of my academic journey. To my brother and sister - thank you for your love and life-long friendship. And to Brian - your love, patience and belief in me kept me whole through all the inevitable challenges I faced.

Zoe Perkins

Boston University

Mechanical Engineering Department

ON THE DEVELOPMENT OF SELF-PACED TREADMILL SYSTEMS FOR GAIT RESEARCH

ZOE PERKINS

ABSTRACT

Self-paced treadmill (SPT) systems use real-time feedback control to dynamically adjust treadmill speed in response to a user's movements, enabling the simulation of overground walking and running in controlled laboratory settings. With the potential to improve the translational and ecological validity of walking and running studies, these systems are uniquely valuable for gait research. However, adoption of SPT systems remains limited - commercial systems are prohibitively expensive, while custom development demands extensive time and effort compounded by a lack of comprehensive guidance on system design and implementation. To address the gap in guidance, this work distills key information and insights on the development of SPT systems for gait research. It reviews the foundational principles of SPT systems, examines existing implementations, discusses critical design considerations and incorporates practical insights from hands-on development of a custom system. By providing a structured and comprehensive resource on SPT system development, this thesis ultimately aims to facilitate future development of these systems for gait research.

Contents

List of Tables

List of Figures

List of Abbreviations

A/D	.....	Analog-Digital
AP	.....	Anterior-Posterior
COM	.....	Center of Mass
COP	.....	Center of Pressure
GRF	.....	Ground Reaction Force
HAL	.....	Human Adaptation Lab
I/O	.....	Input-Output
LiDAR	.....	Light Detection and Ranging
LTI	.....	Linear Time-Invariant
ML	.....	Medial-Lateral
PID	.....	Proportional-Integral-Derivative
RTCS	.....	Real-Time Control System
RTI	.....	Real-Time Interface
SDK	.....	Software Development Kit
SPT	.....	Self-Paced Treadmill

Chapter 1

Introduction

1.1 Gait Research

Gait is a fundamental human movement that relies on the coordinated function of multiple body systems (Pirker and Katzenschlager, 2017). Through the study of gait, researchers gain critical insights into the underlying mechanisms that regulate how a person walks or runs, including their patterns of muscle activation, joint kinematics and kinetics and energy metabolism, to name a few. Through an increased understanding of gait, both within individuals and across populations, researchers can inform approaches to the diagnosis and treatment of movement disorders, the development of prosthetic devices, and the enhancement of athletic performance. Taken together, these outcomes of studying gait demonstrate the importance of gait research for the promotion of human health and well-being.

Gait research is typically conducted in controlled laboratory settings where movement data can be collected using a variety of specialized equipment (e.g. motion capture cameras, force plates and metabolic carts). While this equipment enables precise data collection, it also places limits on the capture volume in which walking or running can be performed. For example, in-ground force plates can only capture ground reaction force (GRF) data when the user is moving on them, limiting data collection to only one or two continuous strides. As a solution, many gait labs use treadmills, which enable participants to move continuously within smaller capture volumes, facilitating the precise collection of large amounts of continuous gait data for researchers to study. Thus, treadmills have become a popular alternative to having participants walk or run overground in studies of gait conducted in

controlled laboratory settings.

While many researchers use treadmills to simulate overground walking and running, there are several indications that this approach may be flawed and that new approaches are needed. Critically, natural gait is characterized by fluctuations in speed, which is well-known to impact gait mechanics (Fukuchi et al., 2019). A common criticism of using treadmills for simulating overground walking and running is that they can only be operated at fixed speeds. Fixed speed treadmill walking has been shown to reduce self-selected gait speed and/or variability in gait parameters compared to overground walking (Malatesta et al., 2017; Schmitt et al., 2021; Dingwell et al., 2001), indicating that treadmill walking may not elicit natural gait movements. Moreover, executing speed changes on a treadmill through manual adjustments or a verbal cue to the treadmill operator introduces distractions that may inhibit natural walking or running when speed changes are desired and then subsequently performed. In addition, it has long been debated whether the mechanics of treadmill gait and overground gait are equivalent. It has been shown that in an idealized world they are (van Ingen Schenau, 1980). However, some studies have shown significant differences between these two conditions (Alton et al., 1998; Schmitt et al., 2021), while others have shown minimal or no differences (Riley et al., 2007; Dingwell et al., 2001) across a range of gait parameters. Moreover, studies have suggested that even when some gait parameters between overground and treadmill walking are similar, muscle activation patterns, joint moments and powers (Lee and Hidler, 2008), as well as energy consumed (Parvataneni et al., 2009), to achieve these movement patterns are often different. Over several decades, this debate has frequently highlighted a need for further investigation, indicating a general and long-term lack of consensus for the current approach to simulating overground walking and running.

1.2 Self-Paced Treadmills

Self-paced treadmills (SPTs) were developed as an alternative approach to traditional treadmills for simulating overground walking and/or running for gait research. They are a component of a larger SPT system that dynamically adjusts the speed of a treadmill to match a user's pace. SPT systems combine a self-pacing control algorithm with sensors and other hardware to enable users to voluntarily move and change speeds while walking or running on a treadmill without any need to use manual control buttons or give verbal cues to an operator. Instead, users control treadmill speed simply by speeding up or slowing down as they would when moving overground. As a result, SPTs offer a more natural experience of walking or running compared to traditional treadmills and have the potential to improve the ecological and translational validity of gait studies in controlled laboratory settings.

The potential for SPTs to simulate overground walking or running better than traditional treadmills has captured the attention of several human movement researchers. As a result, multiple studies have been conducted to compare kinematics and kinetics while walking or running on an SPT, on a traditional treadmill or overground. In studies comparing SPT and overground walking, it has been shown that users self-select similar comfortable walking speeds (Plotnik et al., 2015; Song et al., 2020; Reneaud et al., 2022; Ray et al., 2018) and have similar muscle activation patterns (Ibala et al., 2019) between the two conditions, indicating the potential for SPTs to enable more natural gait patterns. In studies comparing SPT and traditional treadmill walking, it has been shown that walking on an SPT increases stride-to-stride variability (Choi et al., 2017; Sloot et al., 2014). Taken together, the results of these studies suggest that SPTs may offer a more natural alternative to traditional treadmills for simulating overground movements for gait research.

1.3 Motivation

Though SPTs offer a promising new approach for the simulation of overground walking and running, their use by gait researchers remains limited due to significant challenges in acquiring a SPT system. To date, two main types of SPT systems have been used in gait research - commercial and custom. Commercial SPT systems have been used in several studies (Castano et al., 2023; Castano and Huang, 2021; Cha et al., 2017; Ibala et al., 2019; Kao and Pierro, 2021; Kimel-Naor et al., 2017; Plotnik et al., 2015; Sloot et al., 2014; Theunissen et al., 2022; van der Krogt et al., 2014). However, these systems are costly and often come as part of larger motion capture system packages. Since many researchers are typically already invested in expensive research-grade treadmills and motion capture equipment, purchasing an SPT system is often an impractical option. Alternatively, custom SPT systems have been developed. These systems have been used to measure peak performance parameters such as maximum running speed (Bowtell et al., 2009) and VO2 max (Scheadler and Devor, 2015; Hunt et al., 2018), while others have been employed in efforts to improve virtual reality environments (Souman et al., 2010), clinical analyses of disordered gait (Canete and Jacobs, 2021) and rehabilitation (von Zitzewitz et al., 2007). Though custom system development is a more cost-effective option for many researchers, its feasibility is limited by the non-trivial nature of designing and implementing advanced control systems. Even with the design of many self-pacing controllers having been described in detail, it remains both challenging and time-consuming to implement these control algorithms using software. The implementation of open-source code for self-pacing controllers is not necessarily any easier. Despite a few researchers having released code for their SPT systems to assist other researchers with custom development, implementation can become quickly complicated by incompatibilities between hardware, software and even high-level system objectives. In addition, there is such a wide variety in the approaches taken to develop custom SPT systems that, without any comprehensive guidance summarizing the

approaches to designing these systems and providing practical insights to help streamline implementation, the process can become excessively time-consuming. Thus, the impracticalities of expense with regard to money, effort and time when it comes to purchasing or developing SPT systems, can easily deter researchers from using SPTs in their studies. Without improved access to SPT systems, the potential of SPTs to improve the ecological and translational validity of gait research may go unrealized.

1.4 Purpose

To ease the burden of developing a custom SPT system for gait research and facilitate the adoption of this high impact technology within the larger research community, this thesis presents work that aims to assist future researchers in developing their own SPT systems. Through a distilled review of critical background knowledge, existing SPT systems and key considerations for system development, a comprehensive guide to developing SPT systems is presented. In addition, this thesis presents work on the real-world design and implementation of an SPT system in the Human Adaptation Lab (HAL) at Boston University, adding practical insights to the guidance on developing SPT systems for gait research.

Chapter 2

Foundational Principles of Self-Paced Treadmill Systems

To develop an SPT system, it is essential to have a basic understanding of real-time feedback control and gait mechanics. This chapter provides a brief introduction to these concepts, serving as both an entry-level introduction for those without prior exposure and a refresher within the context of SPT systems for those with it. These concepts provide context for subsequent chapters in this thesis and are an essential component of the comprehensive guidance on developing SPT systems that this thesis presents.

2.1 Real-Time Feedback Control

The vast majority of SPT systems rely on real-time feedback control. A real-time feedback control system is a collection of interacting components that work together in real time to monitor the system output, compare it to a desired reference value and adjust the control input based on the error. A basic block diagram demonstrating this concept can be seen in Figure 2·1. The components of the system include the sensor, controller and the plant, which is made up of both the actuator and the process. During system operation, the sensor monitors the output of the process and compares it to the desired reference value of the system to obtain an error. The error is input to the controller, which contains a model for controlling the system called a control law that essentially states how the system should respond to the error. Once the controller determines the appropriate control signal, it is input to the actuator to effect the desired change in system behavior.

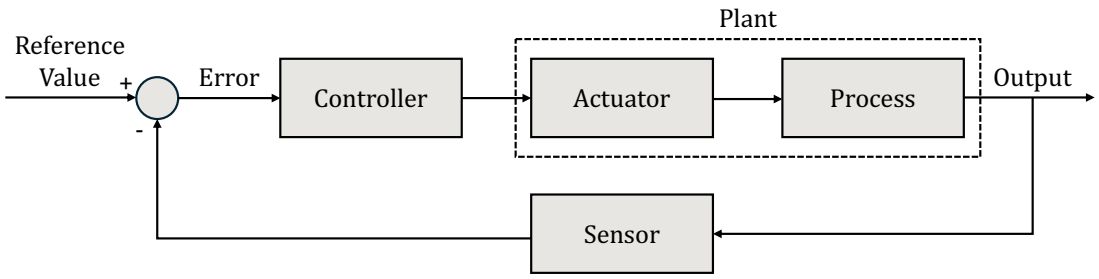

Figure 2·1: Block Diagram for a Basic Feedback Control System.

There are a few specific types of feedback controllers that are of particular importance in SPT systems because they are used so often. These include the Proportional-Integral-Derivative (PID) controllers and observer-based controllers.

2.1.1 Proportional-Integral-Derivative Control

A PID controller directly evaluates the error between the system output and a desired reference value and seeks to minimize this error by producing a control signal that applies three corrective actions to the system. The first action is the P component of the PID controller, which responds in direct proportion to the current error. The second action is the I component, which responds in proportion to the accumulated sum of past errors and compensates for steady-state errors. Specifically, it addresses unknown but constant offsets in actuation that may persist even when the error is close to zero. The third action is the D component, which responds in proportion to the rate of change of the error, aiming to predict and respond to future changes in error to dampen the system's response to rapid changes. Mathematically, the PID control law is written as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt} \tag{2.1}$$

with error $e(t)$, output control signal $u(t)$, time t and proportional gain constants K_p, K_i and K_d to adjust the influence of each of the P, I and D corrective actions, respectively.

In many existing SPT systems, PID controllers are used to compare the user's position on the treadmill to a reference position to produce a control signal that aims to keep the user centered over that position.

2.1.2 Observer-Based Control

Another type of feedback controller frequently used in SPT systems is an observer-based controller. An observer-based controller uses a mathematical model of the system (called an observer) to estimate hidden states of the system. The estimated hidden states are then used in a control law to adjust the system output toward the desired system behavior.

The state of a system refers to the minimal set of variables needed to describe the system's behavior. True states of a system are hidden, meaning they cannot be directly observed or measured. Take, for example, muscle activation (i.e. how much a muscle contracts at a given time). The true muscle activation cannot be directly measured without invasive techniques, but it can be indirectly measured using electromyography, which provides information about the system in the form of electrical activity signals detected as the muscle contracts. Observers, then, enable the estimation of system variables that cannot be directly measured from sensors in the system, including the prediction of future system states. In the context of SPT systems, observers can be used to estimate the user's movements before they occur. When used in a control law, as in observer-based controllers, these estimates essentially allow the SPT system to respond to the user's movements as they are happening instead of afterward.

Observers must be designed. To design an observer, a dynamic model of the system must be defined. For linear time-invariant (LTI) systems, this model can be represented using the general state-space form. Below is the general discrete-time state-space representation of an LTI system:

$$x_k = Ax_{k-1} + Bu_k \tag{2.2}$$

$$y_k = Cx_k + Du_k \tag{2.3}$$

where x_k is the state vector, u_k is the input vector and y_k is the output vector at time step k. A is the state transition matrix describing how states evolve, B is the input matrix describing how the inputs affect the states, C is the output matrix mapping states to outputs and D is the feedthrough matrix mapping inputs to outputs. Once a model for the system has been defined, an observer can be designed. While the details of observer design are not the topic of this work, this brief introduction to observers and dynamic system models lays the foundation for discussing a specific type of observer that is critical to the SPT system design and implementation presented in Chapter 5. That observer is the Kalman filter.

The Kalman Filter

A Kalman filter is a specific type of observer that optimally estimates the states of linear dynamic systems in the presence of system noise and uncertainties (Pei et al., 2019). Thus, a Kalman filter can be particularly useful for estimating states of SPT systems, where sensor measurements are often noisy and the linear dynamics of the user's movements are uncertain. A Kalman filter works by continuously combining a predictive system model with real-time sensor data to refine its state estimates as new measurements become available. The filter explicitly accounts for uncertainty by modeling process noise (uncertainty in the system dynamics) and measurement noise (sensor inaccuracies). At each measurement update, the filter calculates an adaptive Kalman gain, which determines the optimal weighting between model predictions and sensor measurements based on their relative uncertainties. This dynamic adjustment ensures the best possible estimate of the system state. In SPT systems, a Kalman filter can be used to obtain the best possible estimate of the user's position and/or velocity as they move (mostly) linearly on the treadmill in real time.

To further understand how a Kalman filter works, consider the state-space model that

forms the foundation of the Kalman filter's state estimation process:

$$x_k = Ax_{k-1} + Bu_k + w_k, \quad w_k \sim \mathcal{N}(0, Q) \tag{2.4}$$

$$z_k = Cx_k + v_k, \quad v_k \sim \mathcal{N}(0, R) \tag{2.5}$$

In this discrete-time state-space model, x_k is the hidden state vector representing the system's hidden state at time step k, and z_k is the measurement vector containing sensor observations. The hidden state vector evolves according to the state evolution equation (Eq. 2.4), where A is the state transition matrix describing how the state changes over time, B is the control matrix describing how the control input u_k affects the state and w_k is the process noise, which follows a Gaussian distribution with a zero mean and covariance Q. The measurement equation (Eq. 2.5) relates the hidden state to the observations, where C is the measurement matrix that maps the hidden state to the measurement space and v_k is the measurement noise, which follows a Gaussian distribution with a zero mean and covariance R. The Kalman filter uses this model to optimally estimate the hidden system state using a series of alternating computational steps often referred to as the predict and update steps.

At each time step k the Kalman filter uses the state evolution equation (Eq. 2.4) to predict the next hidden state of the system $\hat{x}_k$ and an error covariance matrix $\hat{P}_k$ that quantifies the uncertainty in this state prediction. The prediction reflects a prior belief about how the system is expected to evolve before measurements are made. Computationally:

$$\hat{x}_k = Ax_{k-1} + Bu_k \tag{2.6}$$

$$\hat{P}_k = AP_{k-1}A^T + Q \tag{2.7}$$

Equations 2.6 and 2.7 show how the previous state and error covariance estimations (x_{k-1} and P_{k-1}) are used to compute the current state and error covariance predictions.

When a new measurement arrives from sensors in the system, the Kalman filter in-

corporates this measurement to update the state estimate and computes the Kalman gain K_k to determine how much weight should be given to the measurement versus the model prediction:

$$K_k = \hat{P}_k C^T (C\hat{P}_k C^T + R)^{-1} \tag{2.8}$$

$$x_k = \hat{x}_k + K_k(z_k - C\hat{x}_k) \tag{2.9}$$

$$P_k = (I - K_k C)\hat{P}_k \tag{2.10}$$

Equation 2.8 shows the computation of the standard Kalman gain and equations 2.9 and 2.10 show the computations to acquire updated estimates of the state and error covariance, respectively. The update step combines the prior belief about the evolution of the system state ($\hat{x}_k$) with new evidence of how the system has actually evolved (z_k) to improve the initial estimate of the system state.

In the next iteration of these predict and update steps, the state estimate x_k becomes the previous state estimate x_{k-1} for further computation, allowing the filter to recursively update state estimates in real time using only the information from the prior state estimate and available measurements. As a result, the Kalman filter operates with minimal memory requirements, making it well suited for real-time applications. Moreover, under assumptions of system linearity and Gaussian noise, derivations of the Kalman gain can be shown to minimize the mean squared error of the estimated state, ensuring the filter provides the best possible state estimate (Lacey and Thacker, 1998). This makes the Kalman filter an excellent tool for estimating control inputs for SPT systems.

2.2 Gait Mechanics

In many cases, the inputs needed to control SPT systems, such as user position and velocity, are not readily measurable using available sensors. As a result, many SPT systems rely on sensing the underlying forces and movement patterns that enable coordinated gait to

facilitate calculations of these desired input variables. The study of these underlying forces and movement patterns is called gait mechanics and is critical to understanding how SPT systems work. There are several key types of variables involved in gait mechanics that are of particular relevance to SPT systems. These include the temporal-spatial, kinematic and kinetic variables of gait mechanics.

The temporal-spatial variables of gait mechanics describe measures of time and distance related to gait. These variables include step length, stride length, cadence and gait speed. Of note, gait speed (v_{gait}) can be calculated from step length (l_{step}) and cadence - the reciprocal of step time (t_{step}):

$$v_{\text{gait}} = \frac{l_{\text{step}}}{t_{\text{step}}} = (\text{step length}) \times (\text{cadence}) \tag{2.11}$$

The kinematic variables of gait mechanics include quantities that describe the motion of a person as they move. Human movement researchers are often particularly concerned with the positions of different segments (e.g. trunk, pelvis, thigh, shank, foot) relative to each other; otherwise known as the joint angles. For example, the knee angle is calculated from the relative position of the thigh and shank segment to one another. However, in the context of developing an SPT system, researchers are often more interested in the position and the velocities of these segments relative to an external reference, such as treadmill center.

The kinetic variables of gait mechanics include the GRFs and the moments that can be calculated from them. GRFs are the forces that are exerted by the ground on the foot during gait, helping to externally propel walking and running. They can be used to calculate moments about the coordinate axes of a reference point - typically, the corner of the force plate used to measure the forces. GRFs and moments are often used to calculate the center of pressure (COP) on a force plate, with COP being defined as the point location at which the total GRF force is acting. COP is often used by researchers to detect the position of foot placements on force plates.

In human movement research, gait mechanics are often analyzed and discussed within

the timing of what is called the gait cycle. The gait cycle is a series of repetitive movements that occur during walking (Fig. 2·2) or running. There are a couple of key events in the

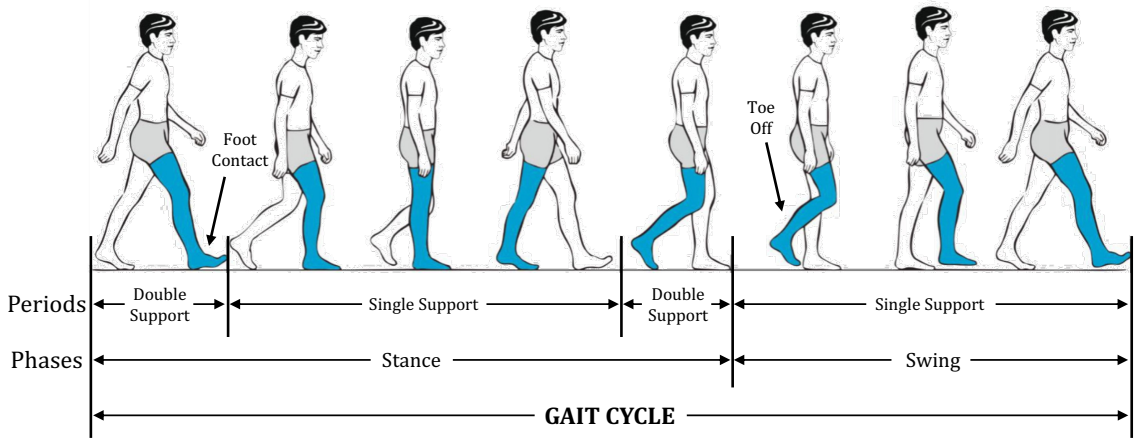

Figure 2·2: Key Periods, Phases and Events of the Gait Cycle for Walking. Adapted from Pirker and Katzenschlager (2017).

gait cycle that are essential to understanding how many SPT systems work. These events include foot contact and toe off. The foot contact event refers to the initial moment of contact between the leading foot and the ground, while the toe off event refers to the initial moment of no-contact between the toes of the trailing foot and the ground. Critically, these events define the transitions between two different support periods that each occur twice during the gait cycle. The double support period refers to time in the gait cycle where both feet are on the ground, beginning at foot contact of the leading foot and ending at toe off of the trailing foot. The periods between double support are called single support, referring to time when only a single foot is in contact with the ground. In addition, there are two phases of the gait cycle called stance and swing, which refer to time in the gait cycle where the foot is either in or not in contact with the ground, respectively. It should be noted here that the gait cycle for running is slightly different than the gait cycle for walking; though foot contact and toe off events remain the same, a period of flight replaces double support, such that the runner transitions between single support and flight periods with no double

support period throughout the gait cycle. These events, periods and phases – foot contact, toe off, double support, single support, flight, stance and swing – are key gait mechanics which are leveraged logically in a number of control algorithms for SPT systems.

Chapter 3

Existing Self-Paced Treadmill Systems

A comprehensive overview of existing SPT systems is an incredibly valuable resource for those who wish to develop their own SPT system. By distilling the variety of existing approaches to the design and implementation of SPT systems, this chapter will help researchers learn from and build on previous efforts, enabling considerable time savings in their own development processes. In this chapter, two key features of every SPT system – self-pacing control algorithms and sensors – are reviewed across a number of existing SPT systems that have been published in the literature.

3.1 Self-Pacing Control Algorithms

A wide range of self-pacing control algorithms for updating treadmill speed based on a user's movements have been previously implemented in SPT systems. Fundamentally, a self-pacing control algorithm aims to calculate a speed update signal (control signal) that adjusts the current treadmill speed up or down. The speed update signal is calculated from the speed update equation (control law), which is formulated from inputs derived from measures and/or models of the user's movement. Most existing self-pacing control algorithms can be generally categorized into one of three groups, though there is often overlap. These groups are: position-based, velocity-based and hybrid control algorithms.

3.1.1 Position-Based

Position-based control algorithms update treadmill speed primarily based on the user's position on the treadmill. The general control strategy involves increasing treadmill speed if the user moves closer to the front of the treadmill, decreasing treadmill speed if the user moves closer to the back and maintaining treadmill speed if the user is at the center. In several position-based control algorithms, this strategy is achieved using feedback control, often in the form of a PID controller (Sec. 2.1.1). The PID error is, then, the difference between the user's position and a desired reference point on the treadmill; often, but not always, at the center of the treadmill. Thus, by minimizing this position error, the PID controller in a position-based control algorithm keeps the user at the desired location on the treadmill.

Classic PID controllers, where all P, I and D components are used, have been implemented in two existing SPT systems, each with additional control mechanisms to further regulate the control signal. In the SPT system by Fruet et al. (2024), the self-pacing control algorithm consisted of a classic PID controller modified by a parabolic gain function of user position. The gain function enabled progressive control of the system's response, where the control signal was increasingly aggressive as the user's distance from the set reference point on the treadmill increased. In a similar fashion, the SPT system by Lichtenstein et al. (2007) used a classic PID controller with an additional dampened zone that attenuated the control signal as a function of the position error when the user was near the center of the treadmill. In the algorithm, the dampened zone logic was added prior to the PID controller and reduced the magnitude of errors that were less than or equal to 10 cm.

In addition to classic PID controllers, several existing SPT systems have implemented PD controllers, which only use the P and D components of PID control to generate a control signal. Both Bowtell et al. (2009) and Minetti et al. (2003) implemented PD controllers with a limited central zone, which ensured the control signal was zero if the user's position

fell within a defined distance of the center of the treadmill, resulting in no speed change. This strategy of dampening or eliminating the control signal near the center of the treadmill has been a common approach in position-based control algorithms for reducing the effect of small changes in speed caused by natural positional oscillations in walking and running that are not necessarily indicative of the user's intent to change speed.

Additional position-based control algorithms that do not use PID or PD controllers have been implemented in existing SPT systems. Several systems have used custom-designed control laws dependent on user position (Hunt et al., 2018; Wang and Hunt, 2023; Reneaud et al., 2022), while another has used an unspecified position-based feedback control algorithm (Choi et al., 2017). The SPT system by Scheadler and Devor (2015) used an algorithm that updated treadmill speed based on eleven discretized treadmill zones: five for acceleration, five for deceleration and one where speed did not change. Each zone was assigned an incremental increase or decrease in speed to be added to the current treadmill speed if the user's position fell within that zone, with increments increasing as distance from treadmill center also increased. An additional control algorithm implemented by Gembalczyk et al. (2020) used an indirect position-based feedback control scheme using fuzzy logic, where treadmill speed updated based on a set of heuristic rules that interpreted user position from the measured swing angle of a rope attached to the user.

3.1.2 Velocity-Based

Compared to position-based control algorithms, velocity-based control algorithms are less common in existing SPT systems. The general strategy in velocity-based control algorithms is to calculate or estimate the velocity of the user from the user's movements and update treadmill speed to match. Four existing SPT systems have used velocity-based control algorithms (Wiens et al., 2019; Canete and Jacobs, 2021; Dong et al., 2011; von Zitzewitz et al., 2007).

In the SPT system by Wiens et al. (2019), a velocity-based control algorithm was used

to calculate the change in a user's walking velocity from changes in user leg swing and center of mass (COM) velocities, both of which were calculated using measures of marker positions obtained using a motion capture system. In the control law for this system, the change in walking velocity was multiplied by a nonlinear gain function that increased the response of the system when the user was closer to the front or back edge of the treadmill. The gain function worked similarly to the position-based control algorithms with dampened central zones discussed in Section 3.1.1. Though the position of the user was considered to determine the necessary gain on the calculated change in user velocity, the algorithm was primarily reliant on the change in user velocity in the control law, causing it to fall into the velocity-based control category.

In the SPT system by Canete and Jacobs (2021), a velocity-based control algorithm was used to calculate user velocity from step time, step length and a kinematic correction to step length that accounted for the distance traveled by the trailing foot during push off. The algorithm performed the velocity calculation at each heel strike, then commanded the estimated velocity to the treadmill with no feedback control involved. Though this procedure was simple, it had the potential to cause a delay between the time the user was actually changing velocity and the time the treadmill speed updated. The developers of this system overcame this potential issue by using large belt accelerations of up to 2.0 ms^{-2} for most steps and 7.0 ms^{-2} for small velocity changes between steps (defined as less than 0.3 ms^{-1}) to match the treadmill speed to the calculated velocity as quickly as reasonably possible.

In other velocity-based control algorithms, rather than calculating real-time user velocity, the *intended* user velocity was estimated with the goal of aligning the timing of the change in the user's velocity with the timing of the treadmill speed update. For example, Dong et al. (2011) performed several regression studies to find a GRF index that linearly correlated with a user's intended walking speed. The index was used in a self-pacing control

algorithm to estimate user walking speed and adjust treadmill speed to match. In addition, the SPT system by von Zitzewitz et al. (2007) estimated the user's intended velocity from horizontal interaction forces measured by a force sensor attached to a belt worn on the user's waist. The algorithm interpreted these forces as the user's intent to change speed. In this system, the user's actual velocity could be measured, enabling the authors to use a PID controller with a velocity error input such that the controller minimized the difference between the estimate of the user's intended velocity and their actual velocity to control treadmill speed.

3.1.3 Hybrid

Hybrid control algorithms combine multiple control strategies into a single control law to produce a treadmill speed update signal. While several of these algorithms rely on some combination of velocity-based and position-based control, there have been many approaches to combining these. Moreover, some hybrid algorithms have used additional control strategies that do not fall into either the velocity-based or position-based control categories.

Several hybrid control algorithms have used an observer (Sec. 2.1.2) to estimate the intended velocity of the user and combined this estimation with position-based feedback control to realize a hybrid control law for updating treadmill speed. For example, Souman et al. (2010) designed an observer that estimated the intended walking velocity of the user based on system measures of the current treadmill speed and user position. The estimate of user velocity, along with the current treadmill speed, measured user position and a reference position on the treadmill, were all input to a control law that output a value for acceleration. The acceleration value was integrated to obtain the updated treadmill speed. Similarly, Wang et al. (2020) used the observer-based methods from Souman and colleagues to estimate the intended walking velocity of the user. However, in their algorithm the estimated velocity was used, along with measured user position, a reference position on the treadmill

and a virtual force, to calculate the acceleration of the treadmill. The acceleration was then used in a PD-like control law to update treadmill speed.

Kim et al. (2012) also designed an observer to estimate the user's intended walking velocity and used the estimate in a control law along with position-based feedback control to obtain an updated treadmill speed. Kim and colleagues additionally introduced a velocity estimation limiter, which reduced aggressive treadmill belt accelerations that occurred in response to large changes in user velocity. The limiter detected large accelerations early using measures of maximum swing foot velocity obtained during the first half of the swing phase of the gait cycle using a motion capture system.

Some hybrid control algorithms have used a Kalman filter (Sec. 2.1.2) to obtain estimates of hidden system states from noisy system measurements. These Kalman filter estimates are used in a control law alongside position-based feedback control to update treadmill speed. In the SPT system by Feasel et al. (2011), user velocity and acceleration were estimated using a Kalman filter from GRF and center of pressure (COP) measurements obtained using an instrumented treadmill. The COP measurements were additionally used to determine the position of the user. The estimated user velocity and position were then used in a control law together to adjust the treadmill speed. The SPT system by Song et al. (2020) was a revision of the algorithm from Feasel and colleagues. Instead of estimating user velocity and acceleration, Song and colleagues estimated user position and velocity. These Kalman filter estimates for user position and velocity were then used in a PD-like control law that updated treadmill speed at every footstep. This rate of update was small compared to that of many SPT systems (30 - 120 Hz) and aimed to reduce the system's response to small changes in user speed and position resulting from natural positional oscillations that occur in walking and running.

Other hybrid control algorithms include those from Yoon et al. (2012) and Ray et al. (2018). In the SPT system by Yoon and colleagues, maximum swing foot velocity was used

to estimate walking speed by assuming the velocity profile of the swing foot was sinusoidal (maximum swing foot velocity was captured using a motion capture system with markers placed on the user's feet). In addition, markers placed on the user's pelvis were used to capture the user's position on the treadmill. Both the estimated user velocity and position were used in a hybrid control law to adjust treadmill speed. In the SPT system by Ray and colleagues, control strategies based on inertial force and user position were combined in a single control law to adjust treadmill speed. The inertial force strategy calculated the body-weight-normalized anterior-posterior (AP) GRF impulse from force plate data captured with an instrumented treadmill. The AP GRF impulse acted as an indicator of the user's intent to push-off the ground and propel themself forward or apply a braking force to slow themself down, acting as a proxy for the user's intent to change speed. In addition, the algorithm used COP measurements to calculate the user's position on the treadmill. The AP GRF impulse and user position were combined in a hybrid control law that adjusted treadmill speed in response to both push-off forces and position error.

3.2 Sensors

A variety of sensors have been used in existing SPT systems to capture measurements of system behavior, which serve as inputs for self-pacing control algorithms. Since most self-pacing control algorithms use control laws that adjust treadmill speed based on the user's position and/or velocity, these sensors are used to capture parameters that can be used to either calculate or estimate these variables. In this section, an overview of sensors used in existing SPT systems, broken down into mechanical, optical, acoustical and magnetic sensor categories, is presented.

3.2.1 Mechanical

Mechanical sensors are a class of sensing hardware that convert physical changes in a mechanical component to an electrical signal that can be measured. In existing SPT systems, these types of sensors have included force sensors, draw-wire encoder sensors and custom mechanical sensors.

Force sensors are some of the most commonly used types of sensors in SPT systems. They contain sensing elements that convert mechanical deformations into voltage. A load cell is a specific type of force sensor commonly found in force plates, a force plate is a specialized platform equipped with multiple load cells and an instrumented treadmill is a specialized dual-belt treadmill equipped with two force plates. Each of these instruments can be used to measure the GRFs exerted by a person standing or moving on the instrument surface. In existing SPT systems that use these instruments (Feasel et al., 2011; Ray et al., 2018; Song et al., 2020; Choi et al., 2017; Dong et al., 2011), measured force data is often used to detect whether the user is in stance and calculate the COPs of the leading and trailing feet at foot contact. These parameters are then often used to calculate an approximate COM or AP hip position of the user on the treadmill as a proxy for user position.

Other instruments containing force sensors have been used in existing SPT systems. In the SPT system by von Zitzewitz et al. (2007), a custom-built measurement instrument comprised of a rigid bar linked to a force sensor linked to a harness worn by the user was employed. The rigid bar kept the user in a fixed location on the treadmill as they walked, and the force sensor measured the horizontal reaction forces produced as the user sped up or slowed down. In this system, the horizontal reaction forces were used to calculate user velocity.

Another type of mechanical sensor commonly used in existing SPT systems is a draw-wire encoder (Bowtell et al., 2009; Hunt et al., 2018; Wang and Hunt, 2023). A draw-wire

encoder measures linear displacement using a retractable wire. To use a draw-wire sensor, the wire is attached to a moving object of interest; in SPT systems, the wire is typically attached to a belt fastened around the user's waist. As the user moves away from the sensor, the wire is pulled, causing a drum within the sensor to rotate. A rotary encoder translates the linear displacement of the wire into a digital signal. The readout of linear displacement often requires very little or no mathematical manipulation to obtain user position.

Custom mechanical sensor technology has also been employed in existing SPT systems. One such technology leveraged an existing body weight support system to measure the swing angle of a rope (Gembalczyk et al., 2020). In this approach, the user wore a harness around their waist and thighs. The harness was suspended by a rope from a trolley-track system installed on the ceiling above the treadmill, and this trolley-track system was equipped with the custom sensor technology. The rope swing angle was used to interpret the position of the user on the treadmill.

3.2.2 Optical

Optical sensors are a class of sensing hardware that detect light and convert it into electrical signals that can be measured. This class of sensor is commonly employed in SPT systems and includes optical motion capture systems, LiDAR and infrared depth sensors.

The most common commercial SPT systems (CAREN, GRAIL, M-Gait, Motek Medical B.V., Houten, Netherlands) have historically employed optical motion capture systems to detect the position of the user on the treadmill. An optical motion capture system is comprised of a set of cameras used to track the three-dimensional coordinate positions of markers attached to a person. The markers themselves can be either active or passive. Active markers emit infrared light that is detected by the cameras, while passive markers are retro-reflective and "bounce" infrared light emitted from the cameras back into the camera lenses. With multiple cameras, the light detected from the markers can be triangulated to produce a 3D marker position. Markers can be placed at specific landmarks on the user's

body such that when the mean marker position is computed, an approximate COM position of the user is obtained. The user velocity can also be obtained by taking the derivative of the mean marker position with respect to time. Additional variables that may be useful for self-pacing algorithm implementation (such as maximum swing foot velocity) can be obtained from marker data. In existing SPT systems that have used optical motion capture systems, markers have been placed at various locations, including around the user's pelvis (Kim et al., 2012), on a helmet worn by the user (Souman et al., 2010) and on the user's feet (Canete and Jacobs, 2021; Wiens et al., 2019; Yoon et al., 2012).

Multiple existing SPT systems have employed Microsoft Kinect devices (Fruet et al., 2024; Wei et al., 2020) or something similar (Kim et al., 2015) to optically detect user position. These devices belong to a class of sensors called infrared depth sensors, which emit pulses of infrared light and measure the time it takes for that light to reflect off an object and return to the sensor (also known as *time of flight*). Based on time of flight and the known speed of light, infrared depth sensors can calculate the distance to objects and provide a depth map of the scenes in front of them. Similarly, Light Detection and Ranging (LiDAR) sensors emit light and use time of flight to calculate the distance to an object. LiDAR sensors have been used in at least one existing SPT system to calculate user position on the treadmill (Reneaud et al., 2022).

3.2.3 Acoustical and Magnetic

Other types of sensors that have been used in existing SPT systems fall into the acoustical and magnetic sensor classes. Acoustical sensors work by detecting and converting sound waves into electrical signals. They work similarly to optical sensors that use time of flight to detect the distance to an object. However, instead of emitting waves of light, acoustical sensors emit waves of sound and use the known speed of sound to calculate distances to objects. One type of acoustical sensor that has been used in existing SPT systems is a sonar sensor (Minetti et al., 2003; Scheadler and Devor, 2015).

Magnetic sensors work by converting changes detected in magnetic fields into electrical signals. At least one existing SPT system has used this kind of sensor to detect the position of the user on the treadmill (Lichtenstein et al., 2007).

Chapter 4

Considerations for Developing Self-Paced Treadmill Systems

There are many different ways to develop an SPT system, as evidenced by the variety of control algorithms and sensors used in existing systems (Ch. 3). However, rather than simplifying the development process, this variety adds complexity. With so many approaches to system design and implementation, how does one select a control algorithm? What about sensors and other hardware? What factors impact these decisions? And ultimately, how do these choices impact research activities and vice versa? These fundamental questions highlight the need for improved guidance in SPT system development.

In this chapter, the discussion of SPT system development moves beyond the foundational principles (Ch. 2) to explore key considerations and facilitate the development of systems tailored to specific needs and constraints. Importantly, the discussion is framed through a translational lens, offering guidance for researchers using these systems for gait research, rather than those focused primarily on advancing control theory in SPT system design.

4.1 Where to Begin?

To begin SPT system development it is imperative to define the intended use of the system. Two key questions to help guide this process are:

1. What is the primary research objective for this system?

2. What population will be studied using this system?

The answers to these questions will help identify critical design goals. For example, if the primary research objective is to study maximum sprinting performance in elite athletes, the system should prioritize a highly responsive control algorithm that adapts to quick changes in speed, along with sensing hardware that minimizes interference with the user's sprinting mechanics. Conversely, if the primary research objective is to study walking in an elderly post-stroke population, the system should prioritize smooth control of speed changes and safety-focused hardware that mitigates fall risk.

In addition, it is equally important to identify critical constraints, as they can limit or shape development choices. For example, coding experience may dictate which control algorithms are feasible to implement or may limit implementation to specific coding languages or platforms. This, in turn, affects sensor and hardware selection based on software compatibility. Similarly, financial constraints can impact the choice of sensors and hardware, which may impose restrictions on control algorithms and overall system design.

To summarize, the development of SPT systems should begin by considering and defining critical design goals and constraints. These factors help narrow the scope of system design, allowing for a more focused evaluation of tradeoffs in control algorithms and sensors, which is the next step of the development process.

4.2 Tradeoffs in Control and Sensing

After defining and assessing critical design goals and constraints, the next step of the development process is the design of the self-pacing control algorithm and selection of system sensors. To facilitate the effective completion of this step, this section presents tradeoffs in control and sensing that should be considered.

4.2.1 Control

To design an effective and safe self-pacing control algorithm, an appropriate balance must be struck between two often competing system goals: responsiveness and smoothness.

In SPT systems, responsiveness refers to how quickly the control algorithm reacts to changes in the user's movements. A highly responsive system reacts almost immediately to the user's movements, minimizing the delay between the user's intent to change speed and the treadmill speed adjustment. While this may seem ideal, excessive responsiveness can be problematic. An overly responsive system may interpret minor variations in gait mechanics or sensor noise as intentional speed changes. The result may be erratic or oscillatory treadmill behavior, with abrupt accelerations and decelerations that destabilize belt movement. This instability makes natural walking or running challenging and introduces safety risks that reduce user comfort and system usability.

Smoothness, in contrast, refers to the continuity of treadmill speed adjustments and aims to ensure a stable treadmill response. A smooth control system prevents sudden speed changes, creating a more comfortable walking or running experience for the user. However, if control is too smooth, the treadmill may respond sluggishly to the user's movements and fail to keep pace with rapid changes in speed. This delay can make the treadmill feel unresponsive and reduce its ability to adapt to the user's movement.

Since the primary goal of self-pacing control algorithms is to respond to a user's movement, many initial designs tend to track the user too closely, resulting in overly responsive system behavior. As a result, much of the subsequent development in self-pacing control has been focused on reducing system responsiveness or increasing smoothness, while still maintaining the sytem's ability to detect and respond to meaningful changes in a user's motion. Several control strategies for achieving this balance have been used in existing SPT systems. They were briefly discussed in Chapter 3 and are summarized in Table 4.1.

Table 4.1: Ways to Balance Responsiveness and Smoothness in SPT Systems.

Method	Primary Effect	Description & Examples
Dampened or Dead Zones	Reduces responsiveness	Implementation of position-based treadmill zones where small gait variations trigger minimal or no speed change (Fruet et al., 2024; Lichtenstein et al., 2007; Bowtell et al., 2009; Minetti et al., 2003)
Limit Belt Acceleration	Reduces responsiveness	Restrict treadmill belt accelerations to prevent abrupt speed changes (Song et al., 2020)
Adjust Control Timing	Balances responsiveness and smoothness	Sync treadmill speed updates with key gait events (Song et al., 2020; Canete and Jacobs, 2021)
Signal Filtering	Increases smoothness	Apply low-pass filtering, Kalman filtering or employ sliding window calculations to smooth noisy input (Song et al., 2020; Feasel et al., 2011)
Improve Velocity Estimation	Increases smoothness	Develop new mathematical approaches to more accurately estimate user velocity from system measurements (Canete and Jacobs, 2021; Dong et al., 2011; von Zitzewitz et al., 2007)

4.2.2 Sensing

In SPT systems, sensor selection involves tradeoffs related to sensor accuracy, sampling rate, cost and interaction with the treadmill user. Among these, sensor accuracy and sampling rate are the two that most directly impact system control. Inaccurate sensor measurements introduce noise to control inputs, limiting system smoothness, while low sampling rates may fail to capture rapid or subtle movements, limiting system responsiveness. Both can lead to unstable system behavior. Generally, higher accuracy and sampling rates come at a higher cost, highlighting the need to consider budget constraints early in the system development process. Consequently, many existing SPT systems developed for gait research rely on sensors that are commonly found in motion capture laboratories, such as instrumented treadmills and motion capture systems, because they have high precision (high accuracy when properly calibrated), high sampling rates and come at no extra cost to the researcher.

Another critical consideration in sensor selection is how the sensor interacts with the user. Wearable sensors such as motion capture markers have the potential to interfere with natural walking or running mechanics. For a researcher whose aim is to study gait mechanics using an SPT system, this *could* be a concern. However, since many gait studies already rely on motion capture markers regardless of SPT use, any minor interference they introduce is generally well-accepted. That being said, it is essential to have a plan for handling situations where a critical marker for system control falls off or becomes occluded. Other wearable sensors, like those attached to belts or safety harnesses, may be more acceptable to some researchers than others. For example, if a study already employs a safety harness, integrating a sensor that attaches to it such as a draw-wire encoder to implement self-pacing control is unlikely to introduce additional constraints, making it a viable option. These tradeoffs highlight the importance of considering sensor selection within the greater context of the primary research objective.

To further guide sensor selection, Table 4.2 presents these tradeoffs in sensor accuracy, sampling rate, cost and user interaction with respect to sensors commonly used in existing SPT systems.

Table 4.2: Tradeoffs of Commonly Used Sensors in SPT Systems.

Sensor	Accuracy	Sampling Rate	Cost	User Interaction
Instrumented Treadmill	Sub-mm COP ($\leq$ 1 mm), sub-N force ($\leq$ 1 N)	Up to 1000 Hz	$\approx$ \$100k–250k	No contact; built-in treadmill
Motion Capture System	Sub-mm marker position ($\leq$ 0.5 mm)	Up to 500–2000 Hz	$\approx$ \$100k–250k	Markers attached to user; occlusion risk
Draw-wire Encoder	Sub-mm distance ($\leq$ 0.1 mm)	100-1000 Hz	\$100-500	Attached to user often with belt or harness
Infrared Depth Sensor	$\approx$ 5-15 mm error	$\approx$ 50 Hz	$<$ \$100	No contact
Sonar Sensor	$\approx$ 10-20 mm error	$\leq$ 50 Hz	$<$ \$100	No contact

4.3 Beyond Control and Sensing

Beyond the design of control algorithms and sensor selection, the development of an SPT system requires careful consideration of additional system components, such as treadmill hardware, control interfaces and host computer. Each of these components plays a vital role in system functionality and must be thoughtfully selected and integrated to ensure safe and effective system performance.

Treadmill hardware, and, in particular, belt length can influence the stability and responsiveness of self-paced control. On treadmills with shorter belts, users can quickly reach the front or back edges of the treadmill platform during large changes in walking or running speed. To counteract this, systems using treadmills with shorter belts often require

control algorithms that react more aggressively to keep the user centered, which can introduce abrupt or uncomfortable speed adjustments. In contrast, longer belts provide more space and reaction time, allowing the system to respond more gradually to user movements, resulting in smoother speed adjustments during large changes in user speed. Several studies have noted the influence of belt length on SPT system behavior. For example, Bowtell et al. (2009) noted that the utilization of an equine treadmill with a belt length greater than 3 m helped users to reach their maximum running speed more easily. Similarly, Kim et al. (2015) wrote that the longer belt length of their custom-made treadmill made implementing self-pacing control easier by reducing the difficulty of keeping the user within the bounds of the treadmill. These findings highlight the importance of treadmill hardware selection in SPT system design, with longer belt lengths generally being considered to reduce the difficulty of implementing self-pacing control.

Control interfaces are responsible for converting control signals into actionable treadmill speed adjustments by communicating with the system's actuators (i.e. the treadmill motors). Some control interfaces are built into the treadmill's hardware and manufacturer-provided software is used to externally control treadmill speed and acceleration from third-party platforms. For example, Bertec treadmills have historically supported TCP/IP communication for external control and newer models now include a Python API for direct programmatic integration. When embedded interfaces are unavailable or insufficient, standalone hardware can be used to establish communication with the treadmill. These interfaces range from high performance real-time control systems (RTCS) like those from dSPACE or Speedgoat (Song et al., 2020), to more cost-effective microcontrollers like Arduino boards (Reneaud et al., 2022; Scheadler and Devor, 2015). While RTCSs offer powerful processing capabilities, they can be complex to configure. However, for developers familiar with MATLAB/Simulink, these systems may be more intuitive than microcontroller boards, which typically rely on C programming. Since the control interface serves

as the critical link between the self-pacing control algorithm and treadmill actuation, failure to select compatible treadmill hardware and control interfaces can prevent the system from functioning altogether. Therefore, careful consideration must be given to the selection and integration of the control interface for successful SPT system development.

Finally, consideration for the selection of a host computer is essential. The host computer of an SPT system must offer sufficient processing power to support real-time signal processing, control and data logging. Moreover, the host computer should be able to support the software environment used for system development and implementation. The overall responsiveness of an SPT system depends in large part on the performance capabilities and compatibility of the host computer with the rest of the system components, warranting consideration.

4.4 Compatibility

The development of an SPT system depends not only on the selection of individual system components but also on ensuring their compatibility with one another. Thoughtful integration of the self-pacing control algorithm, sensors, treadmill hardware, control interfaces and host computer is essential for building a system that is effective, safe and aligned with research goals and constraints.

One of the largest considerations with regard to compatibility in SPT system development is the seamless integration of the self-pacing control algorithm and the selected sensor(s). Some control algorithms and sensors are clearly compatible, while others may be less so. For example, a natural choice of sensor for a position-based control algorithm might be a draw-wire encoder, which provides a direct measurement of the user's position on the treadmill. A force plate, on the other hand, might not seem like an obvious choice for position-based control at first, but by leveraging knowledge of gait mechanics, meaningful position data can be extracted from GRF signals to achieve integration. This example

demonstrates that consideration of how the self-pacing control algorithm will leverage selected sensors to acquire control inputs is essential for effective SPT system development.

In addition to aligning sensor measurements with control variables, achieving compatibility in SPT systems demands consideration of the temporal coordination between system components - that is, how well the system's sensors, processors and actuators exchange data in real time. A critical factor that influences this coordination is latency, which refers to the time delays that occur as data is passed from one system component to the next. In SPT systems, latency arises from three main sources: sensor latency (the delay between a user's movement and the availability of sensor data to the processor), computation latency (the time required to process that data and compute a control signal) and actuation latency (the delay between issuing the control signal and the treadmill's physical response). These latencies reflect not only the timing constraints of individual system components but can also be used to show how the timing behavior of one component can affect that of the next, ultimately shaping the overall performance of the real-time control loop. In other words, because system components are connected in sequence and rely on each other for the performance of their own tasks, a delay in one component can produce delays in the next, propagating delays throughout the system. In control systems where real-time data acquisition, processing and execution are critical to system performance, consideration of the timing of these components is essential. The system's total latency sets a fundamental constraint on how quickly the treadmill can respond to changes in a user's movement at each time step. If total latency is too high, the system response may become delayed, leading to a misalignment between the user's intent to change speed and the treadmill speed adjustment. This can result in system instability. From a compatibility perspective, latency demonstrates the importance of selecting system components whose timing characteristics (e.g. sensor sampling rate, control algorithm update rate, communication protocol type and hardware synchronization capacity) are well aligned.

Finally, although it may seem obvious, all system components must be able to interface with the components that precede and follow them in the connected system sequence. For example, if selected sensors are integrated through an external data collection system to reach the processing hardware, compatibility between the data collection system and the processing hardware must be considered. Failure to consider the compatibility of *all* system components and their interfaces can lead to costly delays, system redesigns or implementation failure.

Chapter 5

Development of a Self-Paced Treadmill System in the Human Adaptation Lab

To develop practical insights and contribute to the guidance on SPT system development established in this thesis, a real-world SPT system in the Human Adaptation Lab (HAL) at Boston University was examined and improved. This chapter reviews both an original and updated SPT system in the HAL and details the work undertaken to implement and evaluate each.

5.1 Background

The Human Adaptation Lab at Boston University is a human movement research lab that aims to elucidate mechanisms of movement that contribute to hip pain. A variety of studies conducted in the HAL have analyzed walking at multiple speeds and emphasized the importance of considering speed when interpreting study outcomes (Lewis et al., 2021; Rao et al., 2024). The outcomes of the HAL's studies often aim to promote the future development of movement-based rehabilitation programs for people with hip pain, making ecological and translation validity of research a key focus of the HAL's work. As the HAL continues to be interested in understanding how speed affects the outcomes of gait analyses and developing real-world rehabilitation programs, there is strong potential for an SPT system to contribute to their work.

The HAL began the development of an SPT system in the spring of 2017 as part of a senior engineering design project at Boston University. Based on the specific interests of

the senior design team, the original system aimed to facilitate analyses of sprinting, which would require treadmill users to be able to quickly reach and sustain maximum running speeds without having to make manual treadmill adjustments that might interfere with running performance. Previous work on the system included the design of a self-pacing control algorithm, selection and set up of system hardware and implementation. While the original system was operational, it was not well suited for walking research with study participants - the primary type of gait research performed in the HAL. When the senior design students graduated, use of the system was discontinued and it was subsequently disassembled over the years. Thus, the design and implementation of an updated SPT system in the HAL was warranted.

As a part of this thesis, work was done to evaluate the original SPT system in the HAL, design a new self-pacing control algorithm and implement an updated SPT system. This work aimed to understand the limitations of the original system and apply new knowledge gained from existing SPT systems to bring the HAL an updated system that meets their needs for future gait research.

5.2 Original System Overview

5.2.1 Self-Pacing Control Algorithm

The self-pacing control algorithm for the original SPT system in the HAL used force plate data from a split-belt instrumented treadmill to calculate the user's AP hip position (hip_y). This position was then input to a PID controller, which adjusted treadmill speed to keep the user at a preset reference point on the treadmill.

Hip Position Calculation

At each time step, the algorithm sampled AP and vertical GRFs (F_y and F_z) and sagittal plane moments (M_x) from both left and right force plates of the instrumented treadmill

at 1000 Hz. The force and moment signals were amplified by scalar gains of 10,000 and 8,500, respectively, and filtered using a 25 Hz low-pass Butterworth filter. Vertical GRFs were monitored to identify foot contact on each force plate, detected when F_z exceeded 50 N. Detection of foot contact was used to identify whether the user was in a period of double support (both feet in contact), single support (one foot in contact) or flight (no feet in contact). During all periods except flight, the AP center of pressure (COP) of the user's back foot ($COP_{y,back}$) was calculated using the force and moment signals from the force plate associated with the side of the user's back foot:

$$COP_{y,back} = \frac{M_{x,back}}{F_{z,back}} \tag{5.1}$$

The AP hip position hip_y was then estimated as:

$$hip_y = COP_{y,back} - L \cdot \frac{F_{y,back}}{\sqrt{F_{y,back}^2 + F_{z,back}^2}} \tag{5.2}$$

where L is the leg length of the user and the fractional term $\frac{F_{y,back}}{\sqrt{F_{y,back}^2+F_{z,back}^2}}$ represents the cosine of the angle between the total sagittal plane GRF (F_{yz}) and its AP component (F_y) (Fig. 5·1). If the user were standing upright, with hip_y directly above $COP_{y,back}$, it would be zero. Scaling the term by leg length gives an approximation of the distance between hip_y and $COP_{y,back}$. Since the sign of the entire second term is negative (as verified by results in Sec. 5.3.2), subtracting it from $COP_{y,back}$ gives the user's hip position relative to the back of the treadmill.

To smooth signal noise and dampen the system's response to small changes in speed resulting from natural positional oscillations that occur during running, computed values of hip_y were stored in a 600-sample-long rolling buffer, from which a sliding window median was computed. If hip_y could not be calculated from $COP_{y,back}$, as during periods of flight, the median value from the most recent stride was added to the rolling buffer instead. Note,

a secondary rolling buffer that reset at the start of each new stride (detected as the transition from flight or double support to single support) stored values of hip_y from the most recent stride.

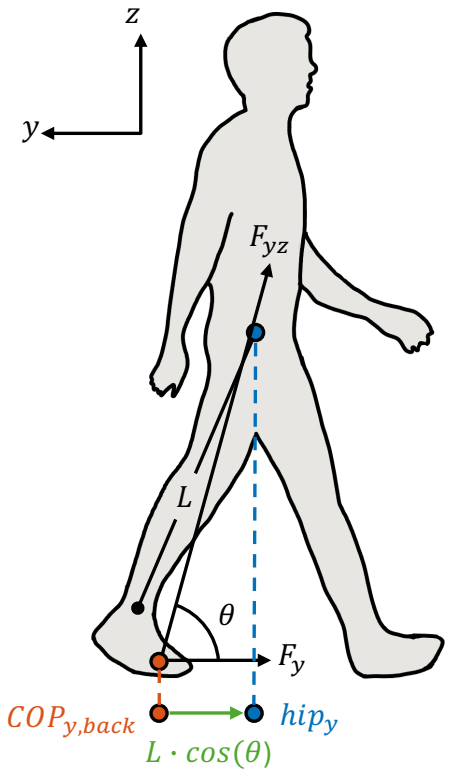

Figure 5·1: Original SPT System: Model for Calculating Median Anterior-Posterior Hip Position. In the original SPT system in the HAL, the AP hip position of the user relative to the back of the treadmill (hip_y) is computed by subtracting the leg-length-scaled cosine of the angle between the sagittal plane GRF vector and its AP component ($L \cdot \cos(\theta)$) from the center of pressure of the user's back foot ($COP_{y,\text{back}}$); see Eq. 5.2.

Treadmill Speed Control

To adjust treadmill speed in response to the AP hip position of the user, the median value of hip_y was input to a PID controller with a preset reference position relative to the back of the treadmill. The controller output $u(t)$ was used to update the velocity of the treadmill

according to the following control law:

$$v_{\text{com}} = \begin{cases} u(t), & |u(t) - v_{\text{curr}}| \leq 0.002 \\ v_{\text{curr}} + 0.002 \cdot \text{sgn}(u(t) - v_{\text{curr}}), & |u(t) - v_{\text{curr}}| > 0.002 \end{cases} \tag{5.3}$$

where v_{com} is the new velocity to be commanded to the treadmill and v_{curr} is the current treadmill velocity. The control law is set up so that at each time step the commanded change in velocity never exceeds 0.002 ms^{-1}. If the PID output is within 0.002 of v_{curr}, the treadmill speed is set to $u(t)$, increasing or decreasing treadmill velocity based on whether $u(t)$ is greater than or less than v_{curr}. Otherwise, the current treadmill velocity is incremented either up or down by 0.002 ms^{-1} in the direction of $u(t)$. The new velocity v_{com} is then commanded to the treadmill with an acceleration of 2 ms^{-2}.

Safety Measures and System Stability

To ensure user safety and maintain system stability, the original self-pacing control algorithm incorporated several control measures designed to prevent falls, ensure smooth speed adjustments and minimize abrupt accelerations and decelerations. One of these was an emergency stopping mechanism, which prevented the user from drifting too far back on the treadmill. If $COP_{y,back}$ came within 0.3 m of the back edge of the treadmill, the system interpreted this as a potential fall risk and immediately brought the treadmill to a halt by setting velocity to 0 ms^{-1} with a deceleration rate of 2 ms^{-2}. In addition, speed constraints were implemented to ensure the treadmill belts always moved in the correct direction (setting the treadmill belt speed to a negative value causes them to move in the reverse direction) and did not accelerate beyond a safe pace. Minimum and maximum constraints on velocity were set at 0 ms^{-1} and 3.5 ms^{-1}, respectively, with these limits having been originally selected for sprinting studies.

Beyond these direct safety measures, the algorithm appeared to be designed to balance

system responsiveness and smoothness to achieve better stability. Several features of the algorithm contributed to this balance, including the selected gain values, signal filtering and the sliding window median applied to the user's hip position. While the original reasoning behind the force and moment signal gains is unknown, they were likely tuned to enhance the controller's response to subtle user movements that indicate an intent to change speed. For example, a slight forward lean could result in a significant change in treadmill speed. This responsiveness might be particularly useful for sprinting but could be destabilizing if not properly controlled. Thus, the low-pass Butterworth filter was likely applied to help smooth signal noise amplified by these gains. The sliding window median filter applied to the user's hip position also appeared to be aimed at increasing system stability. This filter likely improved the treadmill's response to overall movement trends rather than transient fluctuations caused by positional oscillations in natural gait. These features, aimed at balancing system responsiveness and smoothness, further protect the user by preventing abrupt speed changes that could lead to a loss of balance or a fall.

5.2.2 Hardware

The hardware components of the original SPT system in the HAL included a split-belt instrumented treadmill (Bertec Corp, Columbus, OH, USA), a DS1103 real-time control system (RTCS) (dSPACE GmbH, Paderborn, Germany) and a Windows x86 desktop computer (Fig. 5·2).

The split-belt instrumented treadmill is a dual-belt treadmill with a side-by-side force plate configuration. The force plates are installed under each treadmill belt and are capable of measuring GRFs and moments at rates of up to 1000 Hz. A number of hardware components come with the treadmill, including an electronic control unit, two analog converters and a treadmill control unit, interconnected as shown in Figure 5·3. The area suitable for gait on the treadmill surface measures 1619.25 mm long by 609.60 mm wide. The treadmill has a maximum velocity and acceleration of 6.5 ms^{-1} and 30 ms^{-2}, respectively, and

a minimum acceleration of 0.1 ms^{-2}. The treadmill belts can run forward, backward and at different velocities from one another. The Treadmill Control Panel software included with the treadmill can be downloaded on a host computer and used to control belt speed and acceleration. In an emergency, the treadmill can be brought to an immediate stop from both the control panel and an emergency stop button unit placed on the treadmill handrail within reach of the user. Critically, the treadmill control software allows external software applications to remotely control the treadmill via a TCP/IP connection, enabling actuation of the treadmill system from the software where the self-pacing control algorithm is implemented.

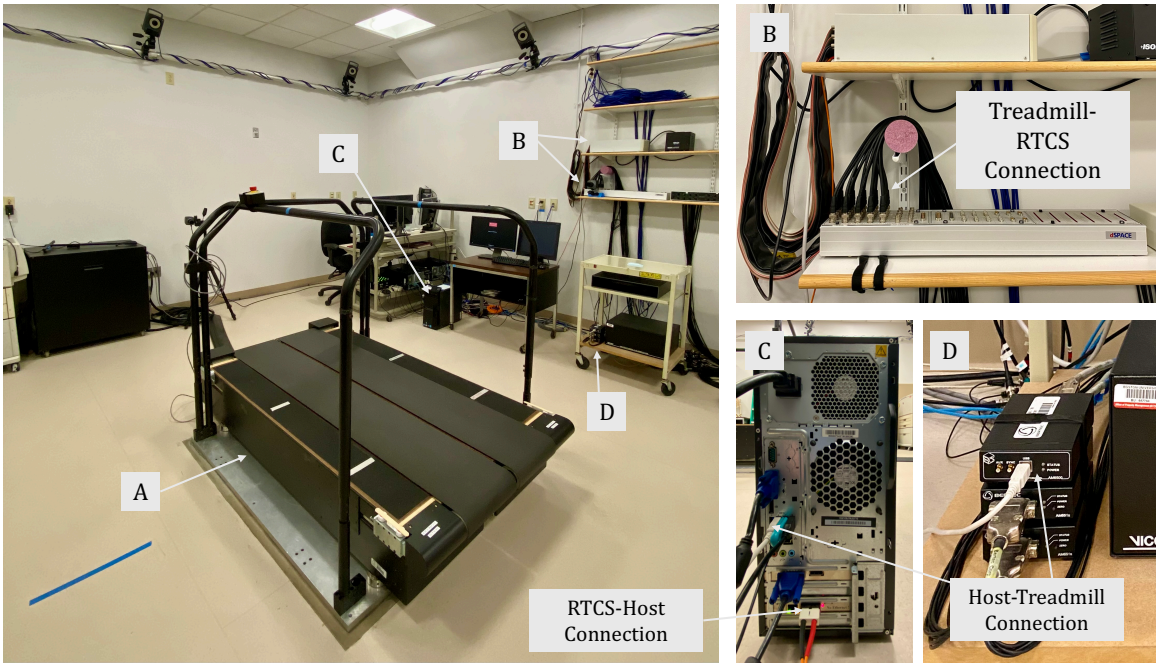

Figure 5·2: Original SPT System: Hardware. The original SPT system in the HAL included (A) a split-belt instrumented treadmill, (B) a DS1103 RTCS and (C) a host computer. A treadmill control unit and two analog converters (D) connected the treadmill to the host computer and RTCS, respectively.

The DS1103 RTCS is comprised of a DS1103 controller board and an input-output (I/O) connector panel, which together enable real-time signal acquisition, processing and control

execution. The controller board serves as the real-time processing unit and is installed in the PCI slot of a host computer, allowing for high-speed data transfer from third-party devices connected via the connector panel. On the host computer, signal data can be monitored and logged using dSPACE ControlDesk software and an implementation software library called dSPACE Real-Time Interface (RTI), which allows integration with MATLAB/Simulink models.

5.3 Original System Implementation

To implement the original SPT system, the hardware components were assembled, the necessary software was configured and initial tests were performed to validate the functionality of various individual system components. After these steps, the original self-pacing control algorithm was fully executed on the system hardware.

5.3.1 Setup

The original system hardware was assembled as shown in Figure 5·3. Critically, to avoid the interruption of ongoing research activities in the HAL, the SPT system was integrated into the HAL's existing data collection system by splitting the treadmill output between the DS1103 RTCS control panel and the HAL's data collection system. In this setup, two host computers were used – one for the SPT system and one for the data collection system. While it might have been possible to use only one host computer by moving the RTCS controller board from its original install location to the host computer belonging to the data collection system, this would have been risky. Given the age of the RTCS, the risk of damaging the board upon removal was high and carried with it the real possibility of no longer being able to implement the original SPT system. It was therefore decided that the controller board would remain in its original host computer. Alternatively, there was the option of moving the data collection system hardware to the computer with the controller

board. However, to be able to run the data collection system hardware from that computer, an operating system update would have been necessary. This option also posed a risk, as there was no certainty about whether the older RTCS controller board would continue to function on an updated operating system. Again, this option risked losing the ability to implement the original SPT system. Thus, in the final integration of the two systems, two host computers were used. Importantly, it should be noted that because the operating system required to run the RTCS with certainty was outdated, the SPT system host computer did not meet the security requirements set by Boston University to be run online. Therefore, due to these constraints, the original SPT system was implemented on a host computer that had to be operated entirely offline and separate from the existing data collection system in the HAL.

With regard to software configuration and control implementation on the host computer, the self-pacing control algorithm for the original system had been previously modeled in Simulink using the dSPACE RTI library and implemented using dSPACE ControlDesk and a custom MATLAB script. Given the amount of time that had passed since the last implementation of the original SPT system and an apparent corruption or loss of files, the dSPACE RTI library was reinstalled on the host computer. The compatibility of the dSPACE RTI library relied on using the 2009a release of MATLAB/Simulink, which was also reinstalled and licensed on the host computer using offline methods. The dSPACE ControlDesk software was already installed and configured appropriately.

5.3.2 Testing

Prior to running the self-pacing control algorithm on the system hardware, the remote treadmill connection, signal acquisition from the treadmill and calculation of median hip position were checked in separate tests to validate their component-level function.

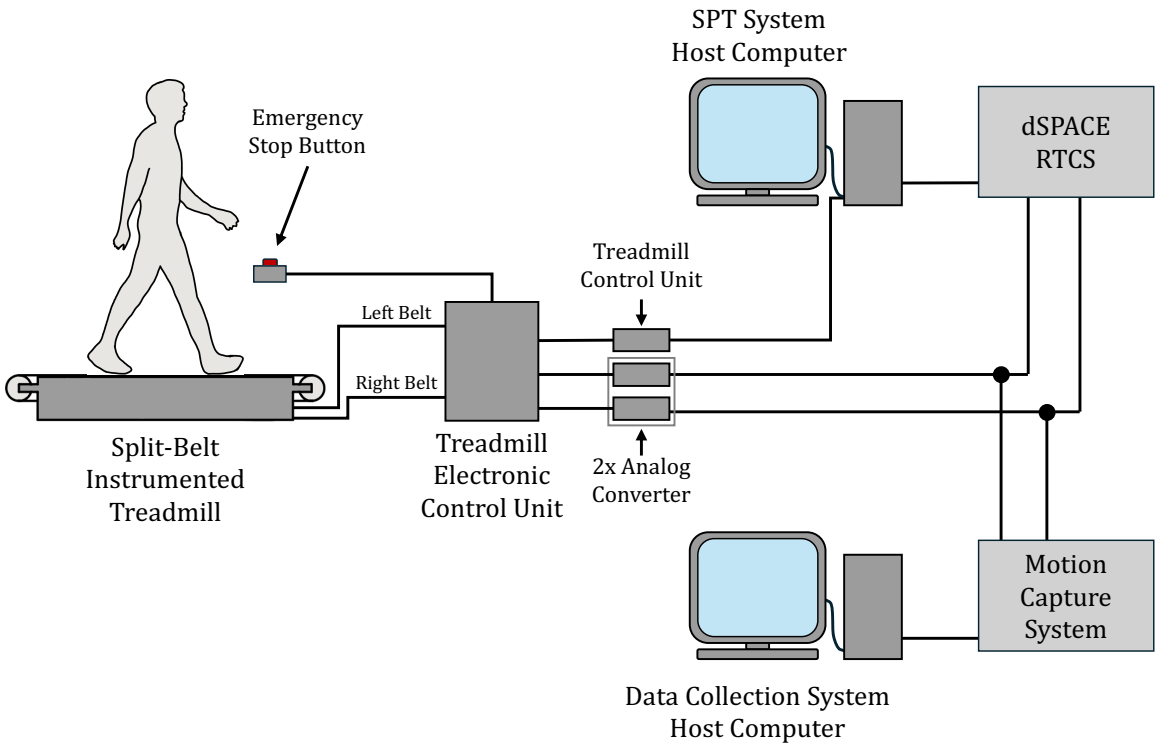

Figure 5·3: Original SPT System: Setup. The original SPT system setup in the HAL was integrated with the existing data collection system, consisting of a motion capture system and its associated host computer. This diagram shows in detail the connections between components in the original SPT system.

Remote Treadmill Connection

To connect, disconnect and send commands to the treadmill, a shared 32-bit C library containing MATLAB callable functions was previously created from manufacturer-provided example code. Critical functions included `initialize`, which established the remote connection between MATLAB and the Treadmill Control Panel software; `setSpeed(speed, acceleration)`, which took values for speed and acceleration and commanded these to the Treadmill Control Panel; and `closeTreadmill`, which terminated the connection between MATLAB and the Treadmill Control Panel software. The functions were run from the MATLAB command window to verify operability and all functions worked as expected.

Signal Acquisition

The acquisition of treadmill signals was verified using the real-time monitoring and logging capabilities of the dSPACE ControlDesk software. After building the Simulink model for the original self-pacing control algorithm, the compiled real-time model was loaded into ControlDesk to establish communication between the software and the DS1103 RTCS. To acquire GRF and moment signals that could be compared to known GRF and moment patterns during walking, a volunteer walked at the approximate center of the treadmill for 10 seconds at a constant speed of 1.25 ms^{-1} while the real-time model was running. As the volunteer walked, the left and right force plate data corresponding to the signal acquisition blocks in the Simulink model were captured and saved to a `.mat` file. No treadmill speed updates were commanded to the treadmill.

To verify that the appropriate signals were captured, the `.mat` file was loaded into MATLAB and each signal, named according to the block names in the Simulink model, was plotted (Fig. 5·4) and compared against known patterns for GRFs and moments during walking (Whittle, 2007) .

Based on the equation used to calculate the AP hip position of the user (Eq. 5.2),

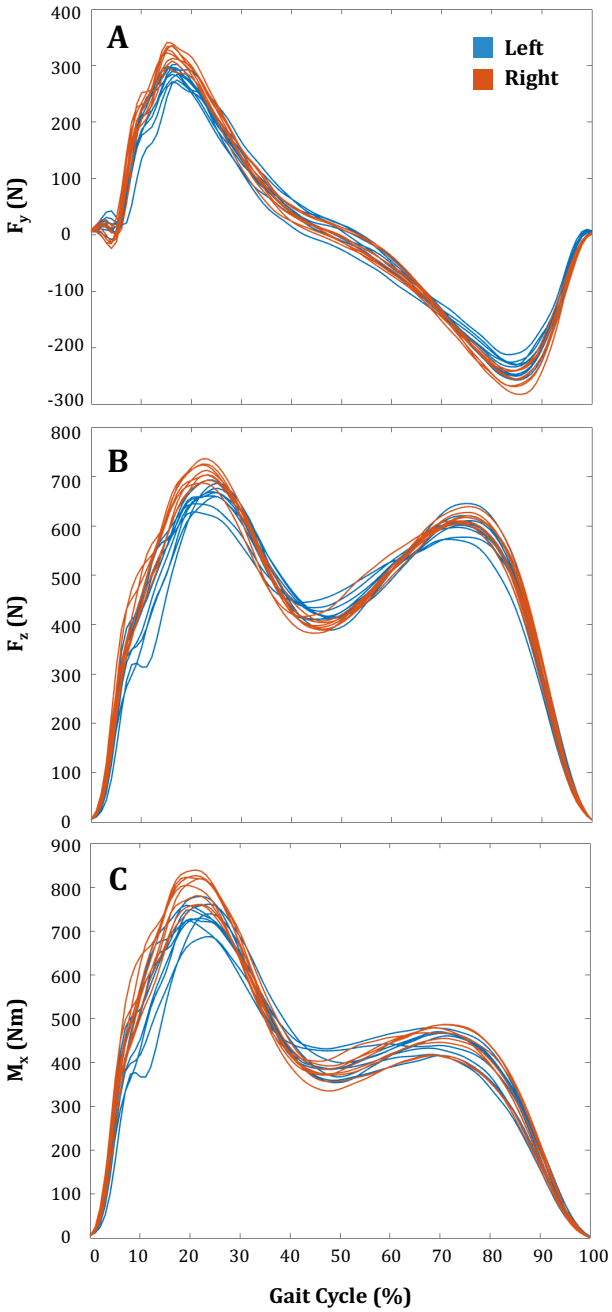

Figure 5·4: Original SPT System: Real-Time Force Plate Data During Walking. AP and vertical GRFs (A and B) and sagittal plane moment (C) captured from the DS1103 RTCS as a volunteer walked on the instrumented treadmill at 1.25 ms^{-1}. Data are normalized to the gait cycle and each line represents data from a single stride.

the sign convention used for GRFs in the original SPT system was determined, where the positive vertical GRF points upward and positive AP GRF points backward. Using this sign convention and known patterns of GRFs during walking, the signals obtained from the RTCS were verified. As expected, the vertical GRFs (F_z) during stance rose at foot contact, fell at toe off and displayed a characteristic two-hump pattern for each stride. Counterintuitively, but aligning with the established sign conventions for the original SPT system, AP GRFs (F_y) were positive at foot contact, crossed over zero at mid-stance and were negative at toe off.

To validate the moment signal, the AP COP (COP_y) was calculated for each foot and plotted from foot contact to toe off (Fig. 5·5). As the volunteer walked approximately at treadmill center ($y = 0.8$ m), the AP COP started in front of treadmill center at foot contact, linearly moved toward the back of the treadmill through treadmill center and ended behind treadmill center at toe off, aligning with the expected pattern of AP COP during walking and validating the moment signals received by the RTCS from the treadmill force plates.

Calculation of Median AP Hip Position

To verify that the calculated value of median AP hip position could be used to track the user's position relative to the back of the treadmill, a volunteer walked on the treadmill, starting at treadmill center. The volunteer then moved forward on the treadmill, backward on the treadmill through treadmill center and back to treadmill center. During the walking trial, treadmill speed was set to a constant 1.25 ms^{-1}. As the volunteer walked, the real-time model ran on the dSPACE ControlDesk software and the calculated median AP hip position was logged to a `.mat` file. A plot of the median AP hip position of the volunteer showed visually accurate tracking of the volunteer's position as they moved on the treadmill (Fig. 5·6). As the volunteer walked at approximately treadmill center, the median AP hip position was shown to be approximately at the center of the treadmill. As the volunteer moved forward, the median AP hip position increased, moving away from the center and

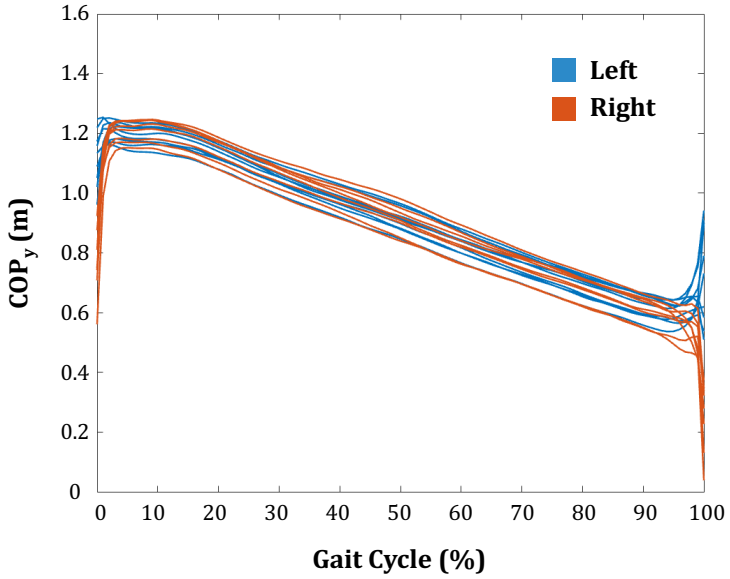

Figure 5·5: Original SPT System: Calculated COP from Real-Time Force Plate Data. Calculated AP COP from force plate data acquired from the DS1103 RTCS while a volunteer walked on the instrumented treadmill at 1.25 ms^{-1}. Data are normalized to the gait cycle and each line represents data from a single stride.

toward the front of the treadmill. As the volunteer moved backward, the median AP hip

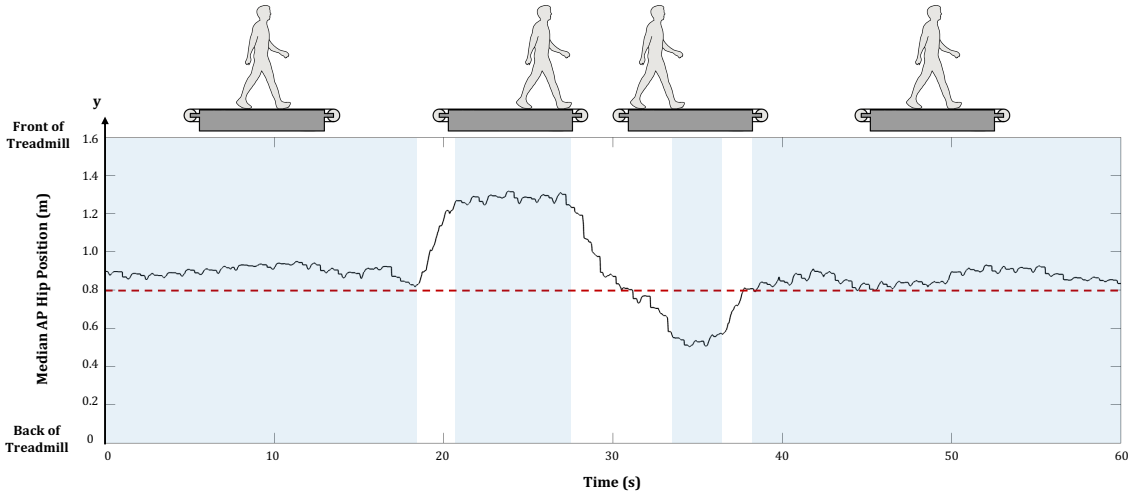

Figure 5·6: Original SPT System: Median AP Hip Position of Volunteer Walking on Treadmill. Plotted data for the test trial described in Section 5.3.2. The volunteer's AP movement on the treadmill, as shown in the pictures above the plot, are reflected in the plotted data. As the volunteer moves forward and backward on the treadmill, their AP hip position relative to the back of the treadmill, as computed by the self-pacing control algorithm, tracks the relative movement of the volunteer appropriately.

position decreased, moving away from the front of the treadmill toward the back of the treadmill, passing through the center of the treadmill along the way. This showed that the calculation of median AP hip position was reliable for tracking changes in user position relative to the back of the treadmill.

Full System Execution

After testing key individual components of the original SPT system, the self-pacing control algorithm was run on the system hardware using a previously written custom MATLAB script while a volunteer walked on the treadmill. Prior to executing the script, the compiled real-time Simulink model was loaded and run in dSPACE ControlDesk and the model outputs were continuously logged to a `.mat` file. The volunteer started from standing at the

middle of the treadmill and the MATLAB script was executed. Upon execution, the script established a remote connection with the treadmill and set the treadmill speed and acceleration to 1.25 ms^{-1} and 0.1 ms^{-2}, respectively, to safely bring the volunteer to a prescribed walking speed and to allow the self-pacing control algorithm to reach steady-state estimations prior to executing self-pacing control. After twenty seconds, the script entered a real-time control loop for self-pacing control. At each iteration of the control loop, the script read the most recently logged file saved from dSPACE ControlDesk, pulled the output velocity command from the PID controller and updated treadmill speed accordingly, while also monitoring the position of the volunteer's back foot for safety. At several points during the execution of the self-pacing control algorithm the volunteer had to grab the handrails attached to the treadmill to maintain their balance as they walked. The volunteer's use of the handrails indicated that the controller might be overly responsive to changes in the user's position, creating unwanted inertial forces that made it difficult for the volunteer to walk naturally. Due to limitations of the original SPT system (further discussed in Section 5.4), additional testing was not pursued.

5.4 Original System Evaluation

The original SPT system in the HAL demonstrated some success in adapting treadmill speed to a user's movement but had significant room for improvement. During full system execution, the treadmill responded to the user's position but did so inconsistently, making it difficult for the user to maintain their balance while walking. Moreover, the self-pacing control algorithm was originally designed for sprinting analyses, and several of its parameter settings, when combined with the unsteady treadmill response, raised safety concerns - particularly with regard to unexpected and abrupt treadmill accelerations. While the safety concerns were partially mitigated by lowering the maximum velocity and acceleration constraints in the control algorithm, resolving the issue of unsteady treadmill response

remained a more complex challenge. Potential solutions included tuning the PID gains, introducing an additional gain function, modifying the timing of treadmill speed updates, incorporating additional control mechanisms or, more likely, some combination of all of these approaches.

Another limitation of the original self-pacing control algorithm was its reliance on leg length as a key parameter for calculating the AP hip position of the user. While leg length measurements using a tape measure are generally considered reliable, their accuracy can vary depending on the individual performing the measurement and the user's posture (Gogia and Braatz, 1986; Cahanin et al., 2024). Additionally, manual measurement is time-consuming and the measured value must be hard coded into the self-pacing control algorithm for each user, adding complexity to the system setup and introducing the potential for error. Incorrect inputs could alter speed adjustments and potentially compromise user safety. Thus, replacing this manual calibration step with a method that relies solely on real-time measurements would improve system usability and protect user safety.

The most critical issue identified in the original SPT system was its outdated hardware. Specifically, the DS1103 RTCS had reached the end of its product lifecycle in 2019 and was no longer supported by its manufacturer. This meant that if the hardware were to fail, no repairs, replacements or troubleshooting assistance would be available. As a result, continued reliance on the DS1103 RTCS posed a significant risk of complete system failure, with no feasible path to recovery, potentially necessitating a full hardware overhaul and extensive redevelopment efforts to restore system functionality. To avoid the possibility of this occurring at a future date, the DS1103 RTCS would need to be removed and replaced in the updated SPT system.

5.5 Updated System Overview

An updated SPT system was designed and implemented in the HAL for use in future gait research. The primary research objective for this system was for it to be used in studies of walking and/or running, and the main populations to be studied were identified as active youth and young adults with and without hip pain. The updated system addressed the key issues identified in the original system. To improve treadmill speed control and reduce system instability, a new self-pacing control algorithm was adopted from work by Song et al. (2020) and modified for use in the HAL. The new algorithm represents a shift from the original self-pacing approach, which relied on a PID controller based solely on user position error, to a more complex implementation of PD-like control using Kalman filter estimates of the user's position and velocity.

During implementation, special attention was given to writing clean and well-documented code to facilitate future modifications and usability for all HAL researchers. Unlike in the original system, the control algorithm implementation in the updated system is written entirely in MATLAB, with no reliance on Simulink – an application not currently used in the HAL. In addition, the DS1103 RTCS was removed. The function of the DS1103 RTCS was replaced with a real-time software development kit (SDK) from Vicon, called Vicon DataStream, that allowed the updated SPT system to operate entirely within the existing data collection system in the HAL, eliminating the need for additional real-time hardware. Taken together, these system improvements not only qualitatively enhanced the performance and stability of the SPT system but also made it more accessible to researchers in the HAL for use in future gait research.

5.5.1 Self-Pacing Control Algorithm

The self-pacing control algorithm for the updated SPT system in the HAL was adapted from a publicly available control algorithm developed by Song et al. (2020). The algorithm uses

force plate data from a split-belt instrumented treadmill and a Kalman filter to estimate the user's position and velocity in real time, then uses these values to update treadmill speed. Here, the algorithm from Song and colleagues is reviewed in detail, and is followed by a description of the modifications made for its use in the updated HAL system, accompanied by the methods and theories underlying these changes.

Overview

The self-pacing control algorithm from Song and colleagues consisted of two key functions: a state estimator and a speed controller. The state estimator employed a Kalman filter to estimate user's position and velocity based on a state-space model of user-treadmill system dynamics and refined these estimates in real time using force plate data from a split-belt instrumented treadmill. The speed controller then used these estimates in a PD-like control law to adjust treadmill speed in response to the user's movements.

State-Space Model of User-Treadmill System. The state estimator in the self-pacing control algorithm defined the system state as:

$$x_k = \begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix} \tag{5.4}$$

where p_{kf} is the Kalman filter estimate of the user's position relative to treadmill center and v_{kf} is the Kalman filter estimate of the user's velocity relative to the lab reference frame. The state evolution equation for the system was defined as follows:

$$\begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_k = A \begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_{k-1} + Bu_k \tag{5.5}$$

where:

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \tag{5.6}$$

$$B = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} \tag{5.7}$$

and u_k is the user's AP acceleration at computational time step k. States from the previous computational time step $k-1$ are used to determine the states for the current computational time step k, Δt represents the computational update interval and AP acceleration u_k is calculated as:

$$u_k = \frac{F_y}{m_{\text{user}}} = a_{\text{mes}} \tag{5.8}$$

where F_y is the AP GRF measured from the system and m_{user} is the user's constant mass. Note, Equation 5.5 should look familiar, as it is a specific formulation of the first equation in the general discrete-time state-space representation of an LTI system introduced in Chapter 2 (Eq. 2.2). The underlying reasoning for formulation of this state evolution equation for the user-treadmill system was understood by substituting A, B and u_k into Equation 5.5:

$$\begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_{k-1} + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} a_{mes} \tag{5.9}$$

By multiplying out the matrices in Equation 5.9, it became clear that the state evolution equation, as defined by Song and colleagues, was simply a discrete-time state-space representation of the linear kinematic equations of motion for the user's position and velocity in terms of the Kalman filter estimates of AP user position and velocity, and the AP acceleration of the user measured from the system:

$$p_{kf_k} = p_{kf_{k-1}} + v_{kf_{k-1}} \Delta t + a_{mes} \frac{\Delta t^2}{2} \tag{5.10}$$

$$v_{kf_k} = v_{kf_{k-1}} + a_{mes} \Delta t \tag{5.11}$$

This confirmed that the state evolution equation used by the algorithm was a reasonable representation of the user's AP linear motion on a treadmill, giving confidence in the use of this algorithm for the updated SPT system.

The measurement equation for the user-treadmill system was defined as:

$$\begin{bmatrix} \bar{p}_{\text{mes}} \\ \bar{v}_{\text{mes}} \end{bmatrix}_k = C \begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_k \tag{5.12}$$

where matrix C was equal to the identity matrix. This equation should also look familiar, as it is a specific formulation of the second equation in the general discrete-time state-space representation of an LTI system introduced in Chapter 2 (Eq. 2.3). This definition by Song and colleagues was accepted as a practical definition of the measurement equation since the measurements of position and velocity derived from the GRFs obtained from the instrumented treadmill, $\bar{p}_{mes}$ and $\bar{v}_{mes}$, directly map to the state variables estimated by the Kalman filter, p_{kf} and v_{kf}.

The process noise covariance matrix Q, measurement noise covariance matrix R and the initial error covariance matrix P_0 for the Kalman filter algorithm were defined by Song and colleagues using empirical walking data. The Q and R covariance matrices were defined as:

$$Q = \sigma_a^2 \cdot BB^T \tag{5.13}$$

and

$$R = \begin{bmatrix} \sigma_p^2 & \sigma_{pv} \\ \sigma_{pv} & \sigma_v^2 \end{bmatrix} \tag{5.14}$$

where σ_a, σ_v and σ_p represent the difference between a_{mes}, v_{mes} and p_{mes} calculated from force plate data and marker trajectory data obtained from the split-belt instrumented treadmill and a motion capture system, respectively. The initial error covariance matrix P_0 was set to the mean of the steady-state values that P reached at the conclusion of multiple pilot runs executed during the system implementation in Song and colleagues' work.

State Estimation. The state-space model of the user-treadmill system as defined by Song and colleagues is used by the self-pacing algorithm to perform the typical predict and update steps of state estimation with a Kalman filter. At each computational time step, the algorithm employs the state evolution equation to *predict* the position and velocity of the

user:

$$\begin{bmatrix} \hat{p}_{kf} \\ \hat{v}_{kf} \end{bmatrix}_k = A \begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_{k-1} + Bu_k \tag{5.15}$$

Note that variables with hats represent Kalman filter estimates prior to receiving measurements from the system. To track uncertainty in these estimates, the algorithm also predicts the error covariance matrix P as:

$$\hat{P}_k = AP_{k-1}A^T + BQB^T \tag{5.16}$$

Note, also, that Equation 5.16 differs from the standard error covariance update equation presented in Section 2.1.2 (Eq. 2.7). Instead, here, the algorithm applies the B matrix to the process noise covariance matrix to account for this noise entering through the control input. This formulation maps the acceleration noise to the state-space, where it can be appropriately interpreted by the system in terms of uncertainty in the position and velocity of the user.

In the same computational time step, vertical GRFs are monitored to detect foot contacts, which are identified when vertical GRFs exceed 20% of the user's body weight:

$$F_z > 0.2 \times 9.81 \times m_{\text{user}} \tag{5.17}$$

Here, the value 9.81 is the acceleration due to gravity on Earth's surface in units of ms^{-2} and is used to translate the user's mass (in kg) into units of Newtons for direct comparison with force values.

If a new foot contact is detected, the algorithm *updates* the Kalman filter state estimate using measurements taken from the system:

$$\begin{bmatrix} p_{kf} \\ v_{kf} \end{bmatrix}_k = \begin{bmatrix} \hat{p}_{kf} \\ \hat{v}_{kf} \end{bmatrix}_k + K_k \left(\begin{bmatrix} \bar{p}_{\text{mes}} \\ \bar{v}_{\text{mes}} \end{bmatrix}_k - \begin{bmatrix} \hat{\bar{p}}_{kf} \\ \hat{\bar{v}}_{kf} \end{bmatrix}_k \right) \tag{5.18}$$

Here, variables with bars indicate mean values between consecutive foot contact detections.

Real-time position and velocity measurements from the system (p_{mes} and v_{mes}) are derived from force plate data as:

$$p_{\text{mes}} = \frac{y_1 + y_0}{2} \tag{5.19}$$

$$v_{\text{mes}} = \frac{y_1 - y_0}{t_1 - t_0} - v_{\text{tm}} \tag{5.20}$$

where y_1 and y_0 represent the AP COP locations of the leading and trailing feet at the moment of foot contact. Trailing foot COP y_0 is calculated by adding the COP of the previous foot contact to the integral of the treadmill speed (v_{tm}) over the time interval between foot contacts ($t_1 - t_0$). The Kalman gain K_k is computed as:

$$K_k = \hat{P}_k(\hat{P}_k + R)^{-1} \tag{5.21}$$

Note that Equation 5.21 is a simplified formulation of the standard Kalman filter update equation presented in Section 2.1.2 (Eq. 2.8), since the observation matrix C is the identity matrix. The error covariance matrix is also updated at foot contact as:

$$P_k = (I - K_k)\hat{P}_k \tag{5.22}$$

The algorithm stores the updated system state, Kalman gain and error covariance matrix in memory for use in the next computational time step. If foot contact is not detected, the algorithm uses the state and error predictions from Eqs. 5.15 and 5.16, respectively, in the next computational time step. Thus, the algorithm estimates states at every computational time step and updates these estimates with measurements from the system at every foot contact.

Critically, the algorithm also monitors the timing between consecutive foot contacts to detect crossover steps. A crossover step is an anomalous gait event in which the user's foot crosses over the midline of a split-belt instrumented treadmill, stepping on the contralateral force plate. Crossover steps create errors in force plate data measurements. As

the performance of Kalman filter state estimates degrades with erroneous system measurement (Truzman et al., 2024), it is important that the algorithm be able to detect these types of steps and remove problematic data from computations. The algorithm by Song and colleagues identifies a previous foot contact as a crossover step when the time between consecutive footsteps exceeds 1.2 seconds. When crossover steps are identified by the algorithm, it skips the Kalman filter update procedure, effectively throwing out the anomalous force data and preserving estimation performance.

Treadmill Speed Control. At each computational time step, the speed controller in the algorithm adjusts treadmill speed based on the Kalman filter estimates of user position and velocity using a PD-like control law:

$$v_{tm,tgt} = \bar{v}_{tm} + G_v \bar{v}_{kf} + G_p(\bar{p}_{kf} - p_0) \tag{5.23}$$

where $v_{tm,tgt}$ is the target treadmill velocity to be commanded to the treadmill, $\bar{v}_{tm}$ is the current treadmill velocity, p_0 is the reference position, $\bar{p}_{kf}$ and $\bar{v}_{kf}$ are the mean Kalman filter estimates of user position and velocity during the user's last step and G_p and G_v are tunable gains on the position and velocity terms, respectively. In addition, treadmill speed is adjusted based on a target acceleration, $a_{tm,tgt}$, calculated as:

$$a_{tm,tgt} = \frac{v_{tm,tgt} - \bar{v}_{tm}}{\Delta t_{tm,tgt}} \tag{5.24}$$

where $\Delta t_{tm,tgt}$ is a preset target time for the treadmill to reach the target treadmill velocity. Both the target velocity and target acceleration are input to the treadmill to actuate a change in treadmill speed.

Modifications

Two algorithmic modifications were made to the self-pacing control algorithm by Song and colleagues to prepare it for implementation in the HAL. These changes included the imple-

mentation of new noise covariance matrices and the addition of a new method for crossover detection. The methods and theories used to accomplish these changes are described below. Additional structural modifications made to the actual code for algorithm implementation are later described in Section 5.6.

Covariance Matrices. To better capture the system and process noise characteristics of the unique SPT system in the HAL, new Q and R covariance matrices were constructed from empirical walking data collected in the HAL following data collection procedures similar to those outlined by Song et al. (2020).

Data were acquired from two volunteers using the HAL's split-belt instrumented treadmill and Vicon motion capture system (Vicon Motion Systems Ltd., Centennial, CO, USA). Four retro-reflective markers were placed on each volunteer's pelvis at left and right, anterior and superior iliac spines. Volunteers walked on the treadmill in eleven walking trials at speeds between 0.8 ms^{-1} and 1.8 ms^{-1}, capturing a wide range of process and measurement noise. Simultaneously, force plate data and marker position data were captured at 1000 Hz from the treadmill force plates and at 100 Hz from the Vicon motion capture system, respectively.

After collection, marker and force plate data were processed. Marker data was gap-filled using Vicon Nexus software (Vicon Motion Systems Ltd., Centennial, CO, USA), then imported into Visual3D (HAS-Motion Inc., Kingston, Ont., CAN) where a low-pass Butterworth filter with a cutoff frequency of 6 Hz was applied and first- and second-time derivatives of marker position were calculated to obtain marker velocity and acceleration. Force plate data were also imported into Visual3D, where they were low-pass filtered using a Butterworth filter with a cutoff frequency of 10 Hz. Cutoff frequencies for filtering were chosen to maintain consistency with typical values used for gait analyses (Crenna et al., 2021). For each volunteer, a structured dataset was created in MATLAB containing the time-series data for marker positions, velocities, accelerations and force plate data from all

walking trials. To align sampling rates, splining was applied to the marker data, increasing it to 1000 Hz. Position, velocity and acceleration data from the four individual markers were combined into single mean values for each variable.

To construct the process noise covariance matrix Q, acceleration residuals σ_a for each walking trial and each volunteer were calculated. Since process noise is relevant to the prediction step of Kalman filtering, residuals were calculated at each computational time step during the walking trials. The mean AP marker acceleration data served as ground truth, and the estimated acceleration was calculated from force plate data using Equation 5.8. Residuals were combined in a single vector and MATLAB's `cov()` function was applied to obtain the process noise variance.

To construct the measurement noise covariance matrix R, position and velocity residuals for each walking trial and each volunteer were calculated at each initial foot contact during the walking trials. The timing of this calculation was chosen based on the fact that the measurement noise is relevant to the update step of Kalman filtering, which occurs only at foot contacts in the algorithm from Song and colleagues. The mean AP marker position and velocity data served as ground truth, and the estimated values were derived from force plate data using Equations 5.19 and 5.20. Residuals were combined into a single matrix, and MATLAB's `cov()` function was applied to obtain the measurement variance and covariance noise.

The new Q and R matrices were then constructed according to Equations 5.13 and 5.14 to obtain:

$$Q = 1.3863 \cdot BB^T \tag{5.25}$$

and

$$R = 10^{-3} \times \begin{bmatrix} 0.3 & 0.5 \\ 0.5 & 2.4 \end{bmatrix} \tag{5.26}$$

Compared to the variance values obtained in Song et al. (2020) these values are of a similar magnitude - a reasonable result given similar system setups and experimental protocols.

To verify that the calculated process and measurement noises could indeed be modeled as Gaussian distributions with zero means, as described by the standard state-space model for Kalman filter state estimation (Eqs. 2.4 and 2.5), the empirical process and measurement noise distributions were plotted (Fig. 5·7). From the plots it can be seen that both process noise and measurement noise are appropriately normally distributed about approximate zero means. With this verification, the covariance values were subsequently updated in the implementation of the self-pacing control algorithm from Song and colleagues for use in the updated SPT system in the HAL.

Crossover Detection. Since accurate force plate data is critical for reliable state estimation in Kalman filtering, an additional crossover detection method was added to the self-pacing control algorithm by Song and colleagues to enhance the system's robustness against such errors. The new method involved a central crossover detection zone defined as a function of the width of the treadmill force plates and a method for determining if foot contacts fell within this zone.

At each foot contact the new crossover detection method calculates the medial-lateral (ML) COP (COP_x) on the force plate of the contacting foot as:

$$COP_x = \frac{-M_y}{F_z} \tag{5.27}$$

where M_y is the frontal plane moment and F_z is the vertical GRF. If COP_x falls within the central crossover detection zone, a crossover step is identified and the update procedure of the Kalman filter is skipped. The new method was incorporated into the implementation of the self-pacing control algorithm from Song and colleagues alongside the existing time-based method for crossover detection.

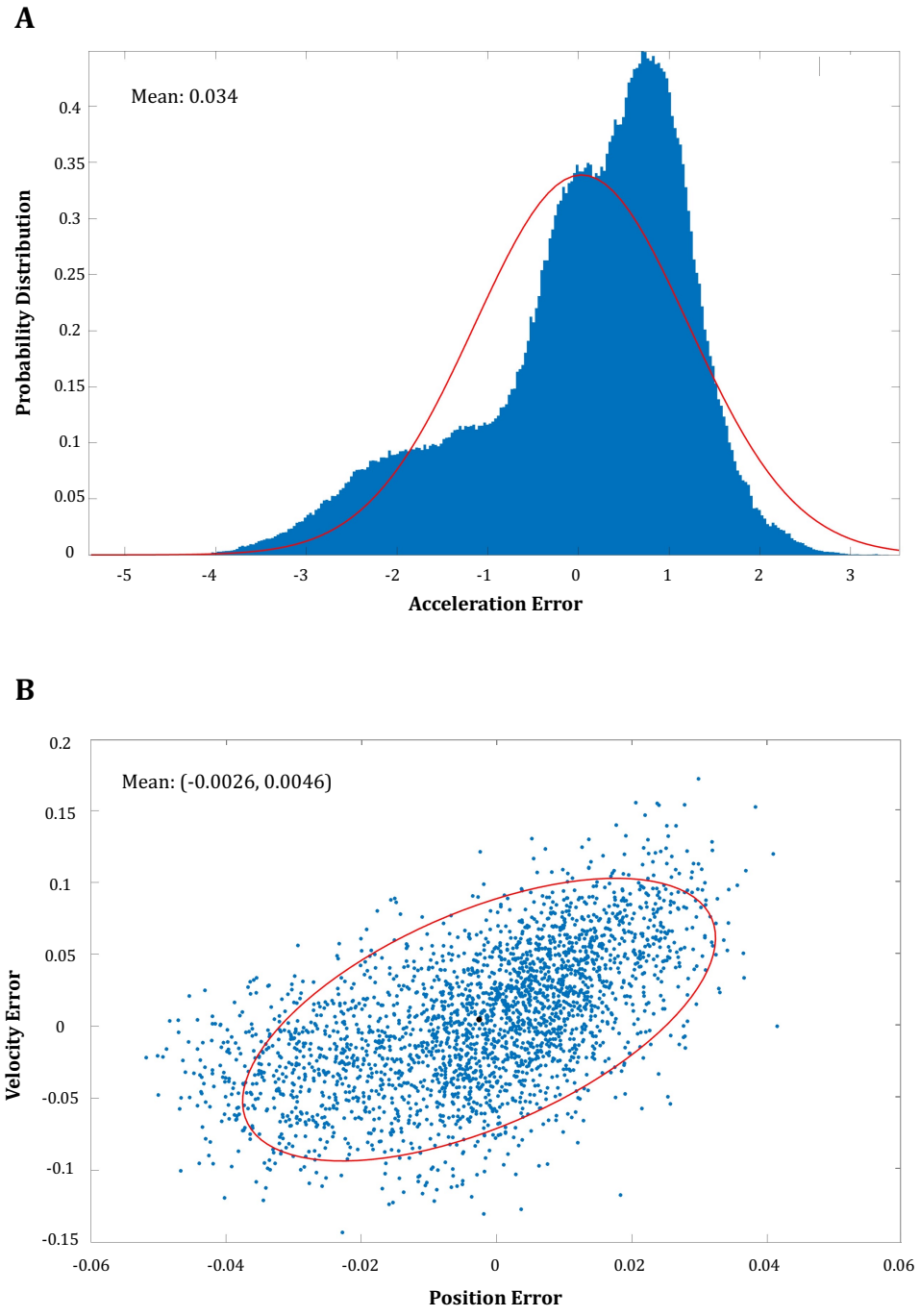

Figure 5·7: Updated SPT System: Empirical Process and Measurement Noise Distributions. A) 1D Gaussian distribution over acceleration residuals and B) 2D Gaussian distribution over position and velocity residuals. Both fits resulted in means close to 0.

Safety Measures and System Stability

In addition to state estimation and treadmill speed control, the self-pacing control algorithm for the updated SPT system in the HAL included several other control features that ensured user safety and maintained system stability. One of these features was an emergency stopping mechanism similar to that implemented in the original SPT system. However, instead of monitoring the COP of the user's back foot, the new algorithm monitored the measured position of the user (p_{mes}), stopping the treadmill if the value fell 0.5 m behind the center of the treadmill. In addition, velocity constraints appropriate for walking and running were implemented and acceleration was limited to avoid abrupt speed changes. The allowable range for treadmill velocity and acceleration commands were 0.001 ms^{-1} to 3 ms^{-1} and 0.001 ms^{-2} to 0.2 ms^{-2}, respectively. Minimum values of 0.001 were selected to maintain consistency with the self-pacing control algorithm from Song and colleagues. Target time to accelerate to a commanded velocity ($\Delta t_{tm,tgt}$) was set to 0.2 s. Though the algorithm by Song and colleagues had set this value to 0.5 s to match the approximate duration of a walking step, it was reduced in the updated SPT system to allow better control of both walking and running speeds. To help with system stability, the algorithm filters incoming force data using a low-pass Butterworth filter and uses the mean GRF and moment signals over the last 100 samples from the force plates in COP computations.

5.5.2 Hardware

In the updated SPT system in the HAL the same split-belt instrumented treadmill as in the original system was retained. The outdated DS1103 RTCS and its host computer were replaced with the HAL's Vicon motion capture system and its associated host computer running a 64-bit Windows operating system. The Vicon system is the same data collection system referred to in Section 5.3.1 and is comprised of 10 motion capture cameras, two digital video cameras, a Vicon Giganet unit, an analog-digital (A/D) interface unit and

integrated third-party devices. The Giganet unit and the A/D interface unit connect to the instrumented treadmill, enabling real-time force plate data collection using Vicon Nexus software. The hardware for the updated SPT system is shown in Figure 5·8. Critically, Vicon Nexus supports real-time device streaming to third-party analysis software using the Vicon DataStream SDK, making it possible to use the Vicon system as a functional replacement for the outdated DS1103 RTCS.

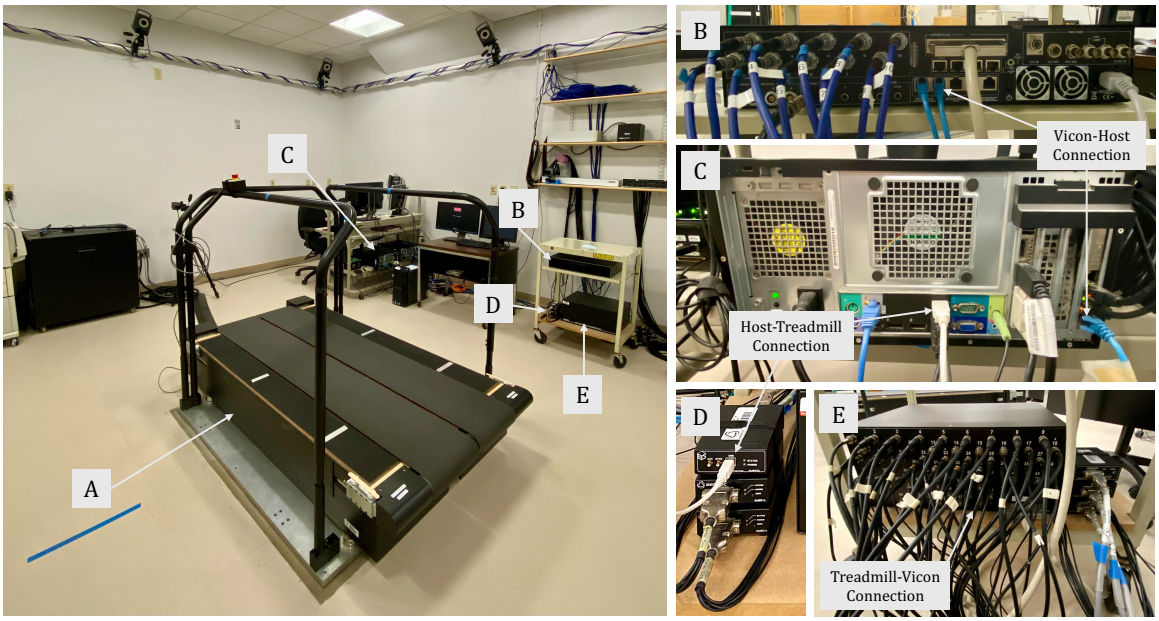

Figure 5·8: Updated SPT System: Hardware. The updated SPT system in the HAL included (A) a split-belt instrumented treadmill, (B and E) Vicon data collection system components and (C) a host computer. A treadmill control unit and two analog converters (D) connected to the host computer and the Vicon data collection system, respectively. The treadmill hardware connected to the Vicon system via an A/D converter unit (E), while the Vicon system connected to the host computer via a Vicon Giganet unit (B). The A/D converter unit and Giganet unit also shared a connection not indicated.

5.6 Updated System Implementation

To implement the updated SPT system in the HAL, all hardware were assessed for proper assembly and necessary software were configured. The open-source code implementing the self-pacing control algorithm by Song and colleagues was downloaded (Song, 2020) and used as a reference for implementing their algorithm with modifications for the HAL using MATLAB. System testing was performed to validate various component-level functions and the updated control algorithm was executed on the system hardware.

5.6.1 Setup

Since the original SPT system had been integrated with the HAL's Vicon data collection system already (Section 5.3.1), only a single hardware adjustment was needed: transferring the treadmill control unit connection from the original system's host computer to the data collection system's host computer. This switch essentially upgraded the entire SPT system from a 32-bit architecture to a 64-bit architecture. A diagram of the updated system setup is shown in Figure 5·9.

The required software for implementation included MATLAB, Vicon Nexus, the Vicon DataStream SDK and the Treadmill Control Panel software. All but the Vicon DataStream SDK were already installed on the host computer. Thus, the SDK was newly downloaded for use in the control algorithm implementation.

To implement self-pacing control for the updated SPT system in the HAL, custom implementation code was written in MATLAB. The code combined elements from both the self-pacing control algorithm implementation by Song and colleagues and the self-pacing control algorithm implementation from the original SPT system in the HAL.

Implementation code written by Song and colleagues was downloaded and evaluated. The code was modeled in Simulink and written to acquire real-time control inputs from a Speedgoat RTCS. To preserve the self-pacing control algorithm logic developed by Song

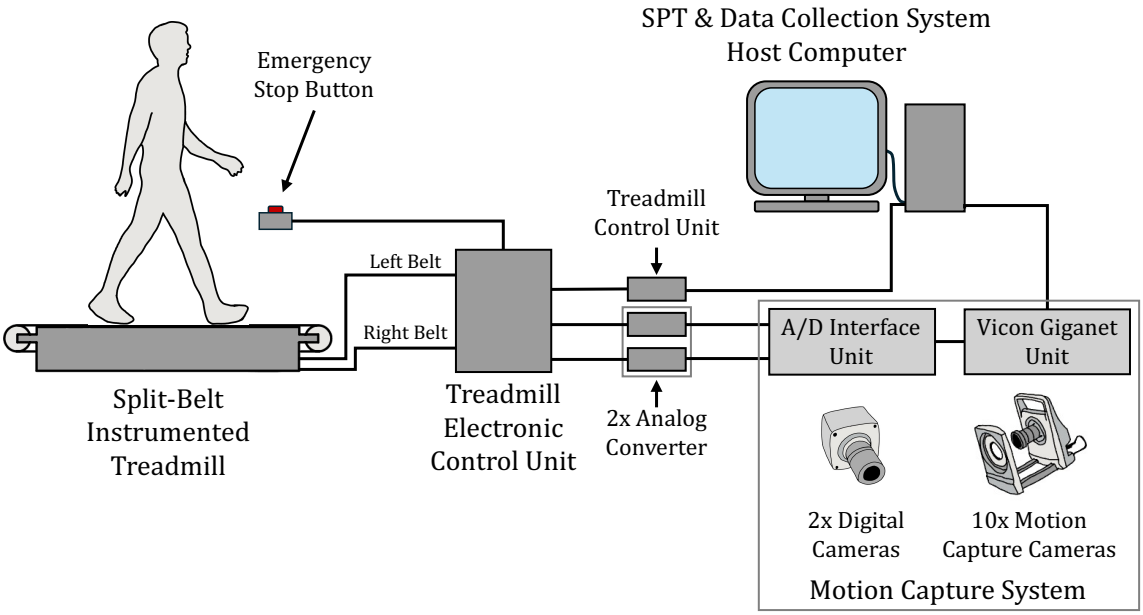

Figure 5·9: Updated SPT System: Setup. The updated SPT system in the HAL no longer relies on the outdated DS1103 RTCS and only has one host computer, which it shares with the existing data collection system in the HAL.

and colleagues, but create implementation code that better suited the needs of the HAL, the Simulink model was restructured as a real-time control loop that could be run from a standard MATLAB script. Key function blocks from the Simulink model, including the state estimation function, the treadmill speed controller function and a function for tracking current treadmill speed, were rewritten as standalone MATLAB functions. Modifications, as described in Section 5.5.1, were incorporated where relevant. These changes resulted in a new script that more closely resembled the implementation code previously written for the original SPT system in the HAL.

To replace the role of the DS1103 RTCS in the original SPT system in the HAL, as well as the same role of the Speedgoat RTCS in the control implementation by Song and colleagues, MATLAB functions leveraging the Vicon DataStream SDK were written. These functions established a connection with the Vicon DataStream server and retrieved AP and vertical GRFs, along with frontal and sagittal plane moments (F_y, F_z, M_x and M_y), from Vicon Nexus in real time. Of note, the Vicon DataStream SDK updates data in the DataStream server at the preset rate of the motion capture system. Since the data collection system in the HAL is set up to capture data at 100 Hz, force plate data captured at 1000 Hz was only available in the DataStream every 0.01 s, in batches of 10 subsamples per frame. Consequently, the real-time loop in the updated SPT system's implementation code was set to iterate every 0.01 s. At each iteration, the code retrieved 10 new force plate samples from the Vicon DataStream server, computed their mean, applied a low-pass filter and input the processed data to the state estimation function.

The implementation code for the updated SPT system in the HAL also used an updated version of the treadmill remote C library to send velocity and acceleration commands output from the treadmill speed controller function to the Treadmill Control Panel software to adjust treadmill speed. This updated library ensured compatibility with the upgraded host computer, having been recompiled from 32-bit to 64-bit using the original source files.

5.6.2 Testing

Remote Treadmill Connection

Since the shared 32-bit C library for connecting remotely to the Treadmill Control Panel software from MATLAB had to be recompiled to 64-bit, the contained functions were once again verified for functionality by calling each from the MATLAB command window. All functions, as previously described in Section 5.3.2, worked as expected on the upgraded host computer using MATLAB 2022a software.

Signal Acquisition

To understand and validate the force plate data signals acquired from the Vicon DataStream, a test similar to that used for verifying signal acquisition in the original SPT system in the HAL was performed. A volunteer walked at the approximate center of the treadmill for 10 seconds at a constant speed of 1.25 ms^{-1} while code leveraging the Vicon DataStream SDK sampled force plate data every 0.01 s. The captured force plate signals were plotted (Fig. 5·10) and compared against known patterns for GRFs and moments during walking.

In the self-pacing control algorithm for the updated SPT system, the positive x, y and z directions were defined as pointing rightward, frontward and upward, respectively. Using this sign convention and known patterns of GRFs during walking, the signals obtained from Vicon Nexus were evaluated. Contrary to what was expected, the vertical GRFs (F_z) during stance fell at foot contact and rose at toe off. While they displayed the characteristic two-hump pattern for each stride, these humps pointed in the negative direction instead of positive. Also unexpectedly, the AP GRFs (F_z) were positive at foot contact and negative at toe off. However, they did crossover zero at mid-stance as expected. Based on this analysis, it was determined that the GRFs needed to be scaled by a factor of -1 to be appropriately used in the self-pacing control algorithm implementation.

To validate the moment signals, the ML and AP COPs (COP_x and COP_y) were calcu-

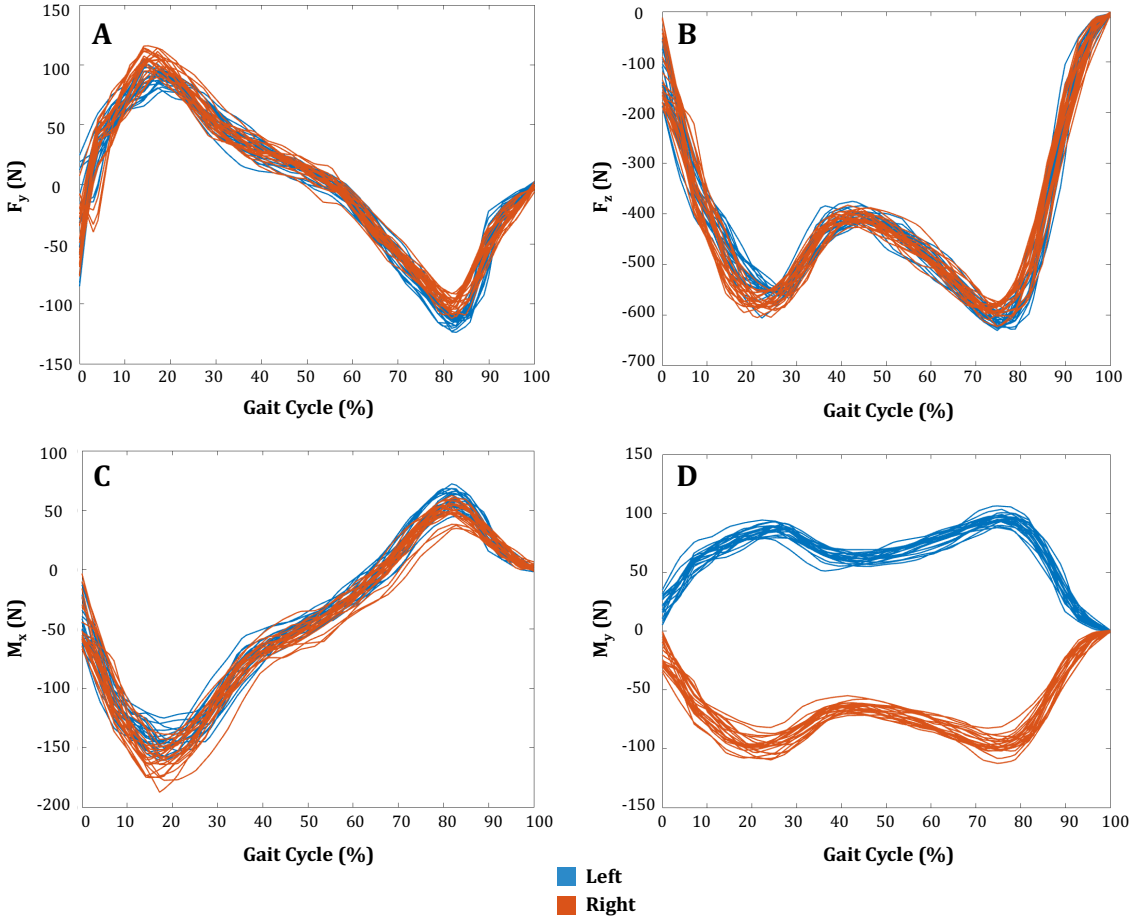

Figure 5·10: Updated SPT System: Real-Time Force Plate Data During Walking. Real-time AP and vertical GRFs (A and B) and sagittal and frontal plane moments (C and D) acquired from Vicon Nexus using the Vicon DataStream SDK while a volunteer walked on the instrumented treadmill at 1.25 ms^{-1}. Data are normalized to the gait cycle and each line represents data from a single stride.

lated for each foot using the *scaled* values for F_z ($-F_z$) and plotted from foot contact to toe off (Fig. 5·11). Using the known appropriate scaled values of F_z allowed for an isolated evaluation of the moment signals from the COP plots. As the volunteer walked at approximately treadmill center (y = 0 m, rather than 0.8 m as in the original SPT system), the AP COP started behind treadmill center at foot contact, linearly moved toward the front of the treadmill through treadmill center and ended in front of treadmill center at toe off; oppositely aligning with the expected pattern of AP COP during walking. The ML COP was also the opposite of what was expected, with foot contacts on each force plate shown closer to the outer edge of the treadmill belts, rather than closer to the inner edge as was expected. Therefore, it was determined that the moment signals also needed to be scaled by a factor of -1 to be appropriately used in the self-pacing control algorithm implementation, as this would flip the COP data to the expected pattern.

From the plots and in line with the same reasoning as presented when evaluating force signals from the original SPT system (Sec. 5.3.2), it was determined that the AP and vertical GRFs needed to be negated in order to be consistent with the positive AP axis pointing forward and the positive vertical axis pointing upward as defined in the self-pacing control algorithm implementation. This was also determined to be the case for frontal and sagittal plane moments. Therefore, the implementation code for the updated SPT system scaled *all* force plate signals by a factor of -1 prior to using them as input for the state estimation function.

Crossover Detection

To test the new method for crossover detection discussed in Section 5.5.1, a volunteer walked on the treadmill while the self-pacing control algorithm was running. To ensure only the new method for crossover detection was evaluated in this test, the crossover detection method from Song and colleagues was not used. During the test, the volunteer walked on the SPT at a comfortable speed and performed 20 steps each at a comfortable step width,

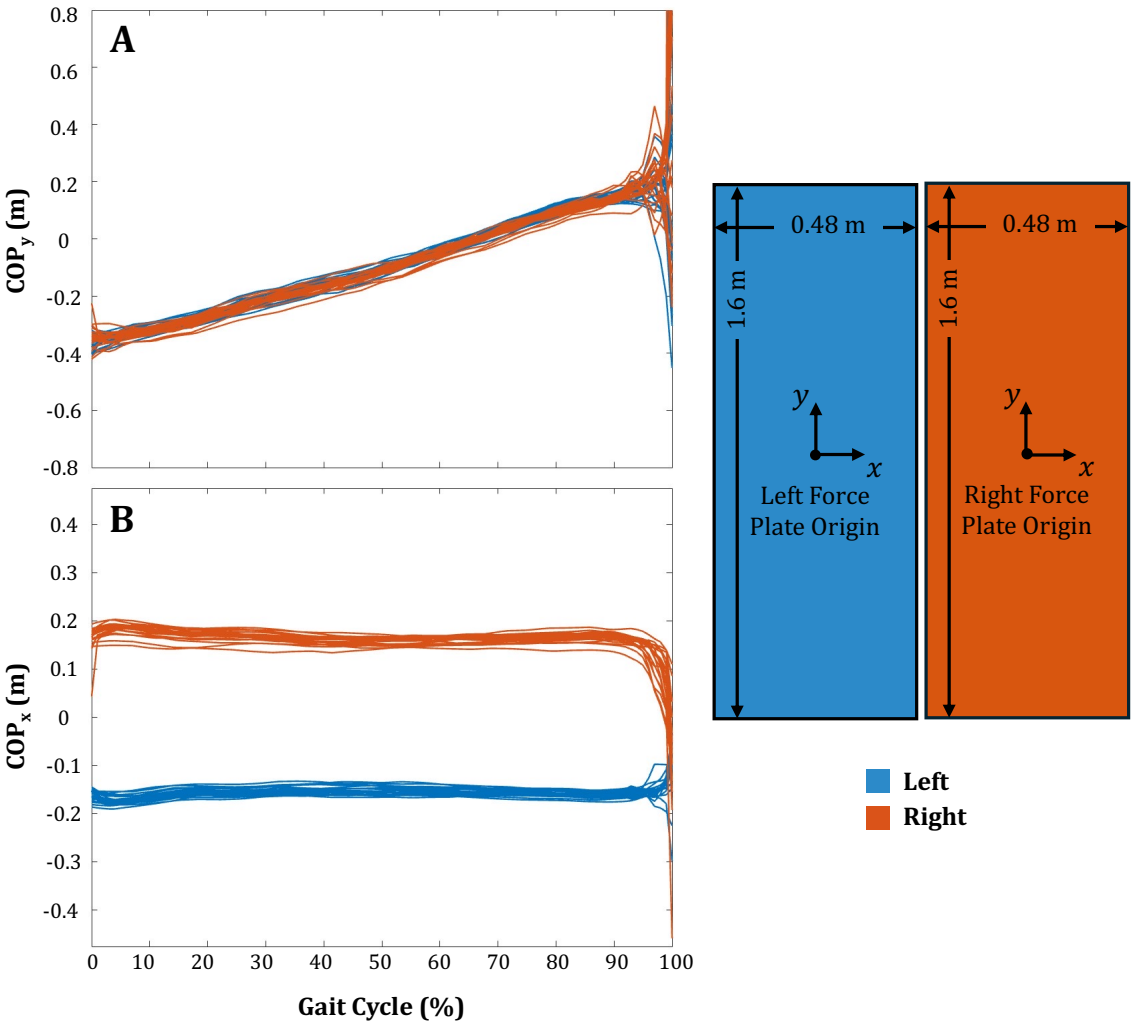

Figure 5·11: Updated SPT System: Calculated COP from Real-Time Force Plate Data. Calculated AP and ML COP (A and B) from force plate data acquired from Vicon Nexus using the Vicon DataStream SDK while a volunteer walked on the instrumented treadmill at 1.25 ms^{-1}. Data are normalized to the gait cycle and each line represents data from a single stride. COP values for each leg are relative to their corresponding force plate origin as defined in Vicon Nexus.

at a narrower step width and at a wider step width. The volunteer aimed to place the narrower steps in the crossover detection zone. For each step detected by the algorithm, the ML COP was calculated (Eq. 5.27), recorded and plotted along the ML axis of each of the treadmill's force plate platforms (Fig. 5·12).

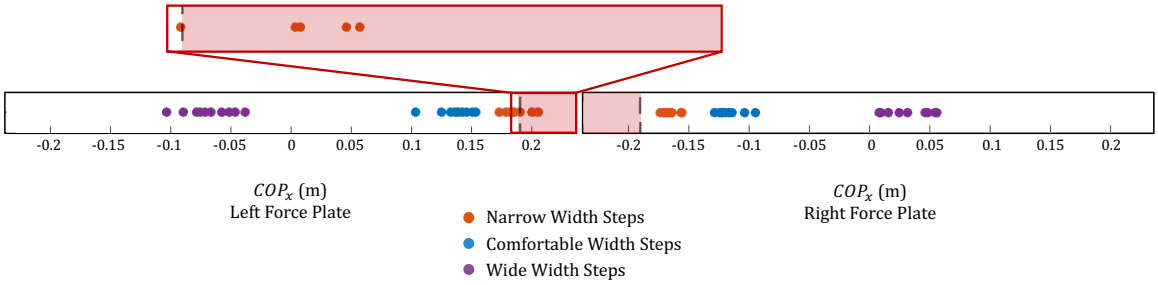

Figure 5·12: Updated SPT System: Crossover Detection Test. ML COP of the 59 steps detected during the crossover detection test described in Section 5.6.2. The crossover detection zone on each treadmill platform is shown in red. Of the 19 narrow steps detected by the self-pacing algorithm, four were flagged as crossover, as shown by the four points to the right of the dashed line in the zoomed in portion of the left force plate.

Of the 60 steps taken by the user during the test, 59 were detected by the algorithm. The step that was not detected belonged to the narrow step width group and was likely missed due to crossover preventing the vertical GRF from exceeding the force threshold required for foot contact detection. Of the 19 narrow steps, only four were flagged as falling within the defined crossover detection zone (within 0.0475 m of either force plate's inner edge). One possible explanation is that although the volunteer's foot appeared to land within the crossover detection zone, the calculated ML COP did not, possibly due to balance-related lateral shifts or a toe-out foot angle at the detected foot contact. This result suggests that the new crossover detection method might benefit from an expanded detection zone. However, expanding the crossover detection zone may risk overlap with the more comfortable step width region, resulting in false crossover detection.

The evaluation of wide steps was also conducted, as the implementation of the new

crossover detection method in the self-pacing control algorithm specifically evaluates whether the absolute value of the ML COP is above 0.19 m. For very wide steps, the absolute ML COP values could theoretically fall within the crossover zone. However, the data for wide steps show that the ML COP values fall at sufficient magnitudes to avoid false crossover detection. Though this data is only representative of a single volunteer, if the volunteer had performed even wider steps, they would have stepped off the moving treadmill belt and onto the stationary force plate platform. Thus, the wide step data effectively shows the safe upper limit on step width. If the user were to step beyond this limit and off the moving belt, crossover detection would be a much lesser concern compared to the reason the user stepped off the belt.

Importantly, it should also be noted that during subsequent crossover detection testing, narrow steps involving crossover sometimes triggered the position-based safety mechanism designed to stop the treadmill if the user's position was too close to the back of the treadmill. It is likely that erroneous force plate data arising from the crossover steps distorted the AP COP calculations, leading to miscalculations of the volunteer's position and falsely activating the safety threshold. This observation underscores the importance of detecting crossover accurately, but also suggests that the safety mechanism may require a more robust method for crossover handling. Consequently, for the remainder of system testing, volunteers were advised to avoid crossover steps.

Full System Execution

To execute the updated SPT system, the new implementation code was run with a volunteer walking on the treadmill. The volunteer started from standing at the approximate center of the treadmill before the code was executed. Upon execution, the system established a remote connection with the treadmill via functions from the treadmill remote C library and initialized a client connection with Vicon Nexus using the Vicon DataStream SDK.

Before initiating treadmill movement, the system captured the vertical GRFs of the

volunteer as they stood still on the treadmill and used these signals to compute the user's mass (m_{user}). As in the original SPT system, the treadmill was initially commanded to accelerate at a rate of 0.1 ms^{-2} to a constant speed of 1.25 ms^{-1}, where it stayed for the remainder of a 10 second initialization period. This method was used to ensure a safe transition to walking and allowed the self-pacing control algorithm time to establish steady-state estimation before entering the real-time self-pacing control loop.

After the initialization period, the system transitioned into the real-time self-pacing control loop, iterating approximately every 0.0135 s. In each iteration of the control loop, the following functions were called (for simplicity, function inputs and outputs are not shown):

1. `toc`: Retrieves the time elapsed since the start of the implementation code.
2. `trackTreadmillSpeed`: Determines the current velocity of the treadmill.
3. `getGRF`: Retrieves 10 new samples of force plate data from Vicon Nexus, computes their means, applies a low-pass filter and scales signals by -1.
4. `stateEstimator`: Estimates user position and velocity using a Kalman filter and measured force plate data.
5. `calcTargetTreadmillSpeed`: Calculates target treadmill velocity and acceleration from Kalman filter state estimates.
6. `setTreadmillSpeed`: If position of user is not close to the back of the treadmill ($>$ 0.5 m behind treadmill center), sets treadmill speed to the target treadmill velocity; otherwise, halts treadmill belts.

During this execution of self-pacing control the volunteer appeared to walk comfortably and have effective control of treadmill speed. The volunteer successfully increased the treadmill speed from a walking pace to a jogging pace and then slowed it back down.

Additionally, when the volunteer intentionally allowed themself to drift toward the back edge of the treadmill, the position-based safety mechanism appropriately halted the treadmill belts when the measured user position fell 0.5 m behind the center of the treadmill.

State Estimation and Control of Treadmill Speed

To analyze the performance of Kalman filter state estimation and evaluate the impact of each state estimate on treadmill speed control, key variables from the state estimation and treadmill speed controller functions were recorded as a volunteer walked on the treadmill. In this trial, the volunteer started from standing at the center of the treadmill. Upon code execution, the treadmill accelerated at a rate of 0.1 ms^{-2} to a constant speed of 1.25 ms^{-1}. Once the system entered the real-time self-pacing control loop, the volunteer was walking comfortably at the approximate center of the treadmill. They were then instructed to move toward the front of the treadmill and return to treadmill center twice.

Throughout the trial, Kalman filter estimates and measured values for user position and velocity were recorded, along with the current treadmill velocity as tracked by the control algorithm. The target treadmill velocity as calculated by the treadmill speed controller function (`calcTargetTreadmillSpeed`) was recorded as well. Kalman filter estimates of user position and velocity recorded throughout the trial were plotted separately along with their measured values obtained from force plate data (Fig. 5·13). A separate plot with the Kalman filter estimates, current treadmill velocity and target treadmill velocity is also shown (Fig. 5·14).

Figure 5·13 shows that the Kalman filter state estimations of user position and velocity appropriately tracked the volunteer's position and velocity as they walked on the treadmill. As the volunteer moved forward on the treadmill from treadmill center, the Kalman filter estimate of user position moved in front of treadmill center (> 0 m). Similarly, velocity relative to the lab reference frame increased as the volunteer moved forward on the treadmill and decreased as they moved back to treadmill center. Additionally, the plots demonstrate

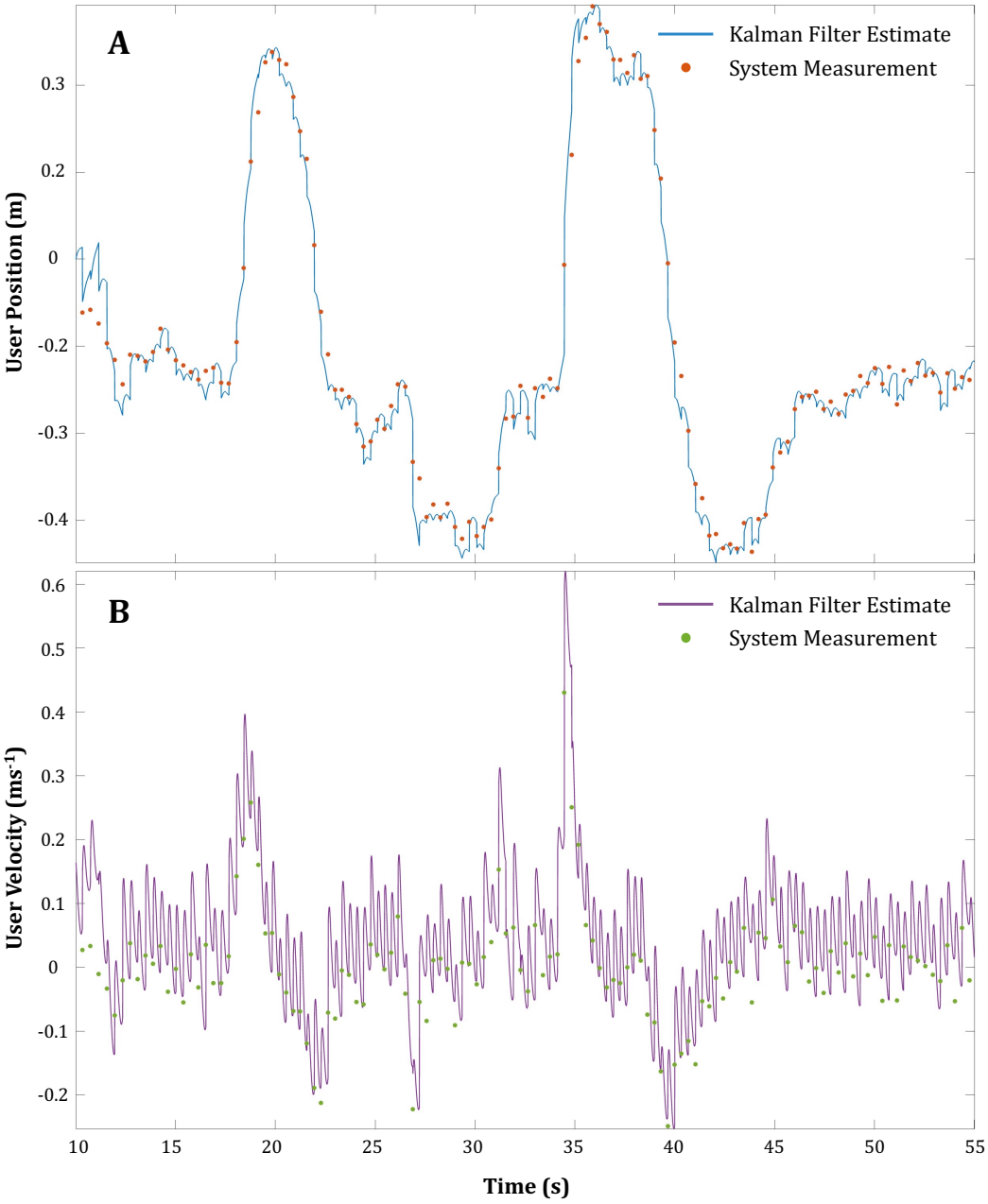

Figure 5·13: Updated SPT System: Kalman Filter Estimates and System Measurements for User Position and Velocity. In this test, the volunteer walking on the treadmill moved forward on the treadmill at approximately 17 s and 34 s, and moved backward on the treadmill at approximately 20 s and 36 s. Position estimates are relative to treadmill center ($y = 0$ m) and velocity estimates are relative to the lab reference frame.

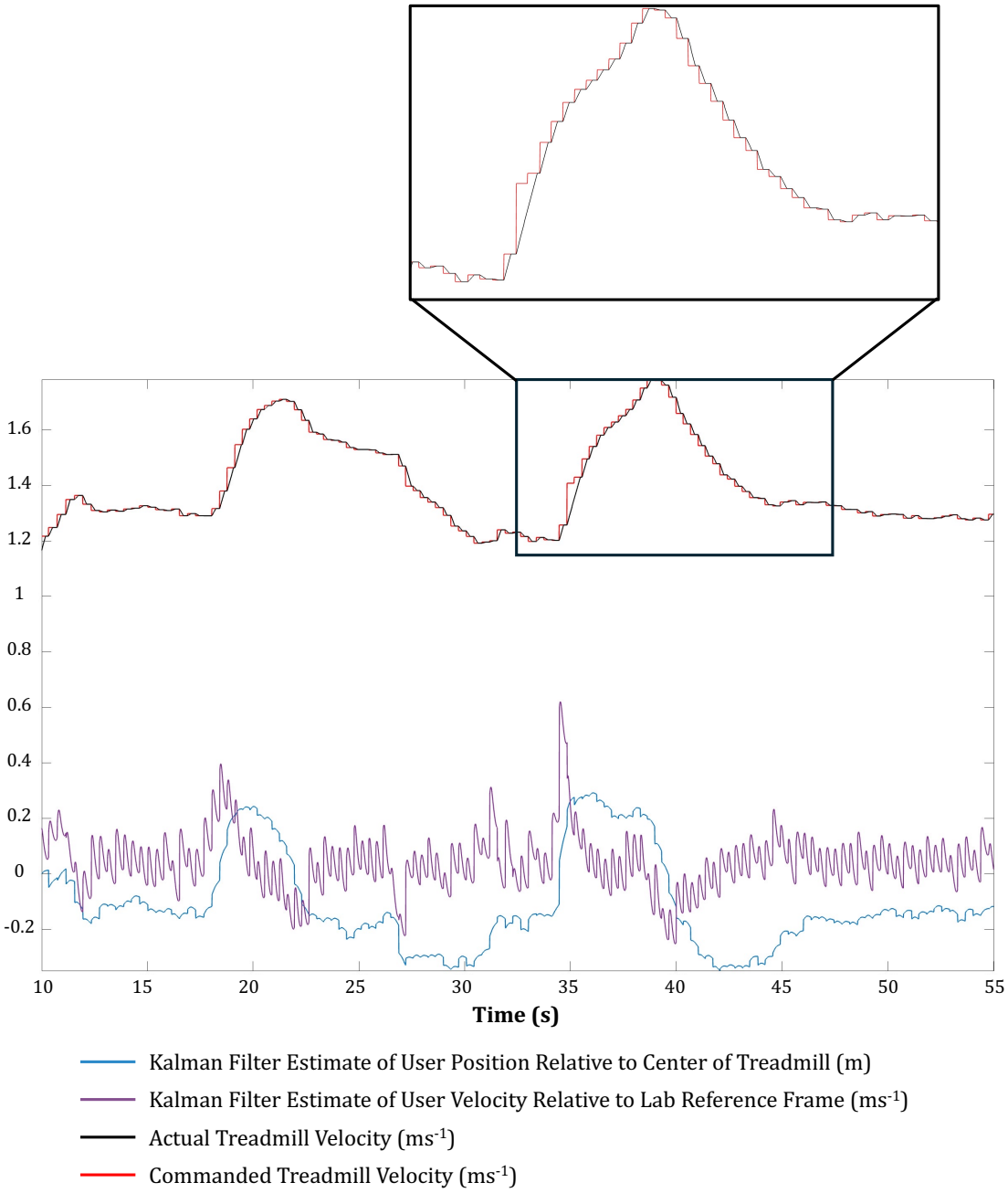

Figure 5·14: Updated SPT System: Influence of Kalman Filter Estimates on Treadmill Speed. As Kalman filter estimates increase and decrease, so too does the commanded treadmill velocity. Though treadmill speed commands are abrupt, actual treadmill speed adjustments are smoother, as seen in the zoomed in box.

how measurements from the system improve the Kalman filter estimates. When new measurements from the system arrive, the Kalman filter state estimates are shown to abruptly jump toward the measurement value (seen as a sharp jump in the continuous Kalman filter estimate toward a system measurement dot). The magnitude of the jump is likely dictated by the Kalman gain, which weighs how much to trust the system measure compared to the original prediction.

Figure 5·14 shows the influence of the Kalman filter estimates on commanded treadmill velocity and the resulting actual velocity of the treadmill. The commanded treadmill velocity, resulting from the calculation of the control law in Equation 5.23, follows the same pattern as the Kalman filter estimates; when user position and velocity increase or decrease, so too does the commanded treadmill velocity. Tracking of the actual treadmill velocity showed that treadmill control was both smooth and responsive. Visually, there were no abrupt changes in treadmill velocity and treadmill velocity reached commanded speeds in a timely manner.

5.7 Updated System Evaluation

Compared to the original SPT system in the HAL, the updated SPT system introduced several advancements that enhanced usability for future gait research. Most notably, volunteers walking on the instrumented treadmill with the updated system showed improved control over treadmill speed and reported greater comfort compared to the original system. Critically, volunteers were able to both walk and run comfortably, enabling use of the system for studies of either movement, aligning with the primary research objective for this system.

Though users generally reported feeling comfortable using the updated SPT system, it was noted that speeding up felt easier and more natural than slowing down. This perception may be due to inherent differences in how humans accelerate and decelerate during

real-world overground locomotion. During overground locomotion, increasing walking or running speed typically requires a gradual buildup of momentum to reach faster paces, while slowing down can occur much more quickly. As a result, the sensation that slowing down on the SPT is more difficult than speeding up may reflect a mismatch between the treadmill's self-pacing behavior and a user's expectations of the time it should take to accelerate and decelerate based on their overground walking experiences. In addition, users may feel more cautious when slowing down, as doing so requires moving closer to the back of the treadmill, where there is a greater risk of falling. Thus, the perceived difficulty of slowing down on the SPT may also stem from a user's natural reluctance to approach the treadmill's rear edge. In contrast, speeding up moves the user toward the front of the treadmill, which might feel safer and more controlled because the front edge of the treadmill is usually within the user's line of sight. Thus, the perceived differences between speeding up and slowing down on the SPT may be influenced not only by time-based expectations of acceleration and deceleration based on overground experiences but also by perceived safety on the treadmill.

In addition, the updated system featured significant hardware upgrades, improving longevity and usability. The outdated DS1103 RTCS hardware was replaced with the HAL's existing Vicon data collection system, and the implementation code was redesigned to leverage the Vicon DataStream SDK for real-time force plate data acquisition. This shift in hardware simplified the system's hardware setup and is likely to encourage integration with existing workflows. While the shift in hardware limited real-time force plate data sampling to 100 Hz - down from 1000 Hz in the original system - this reduction did not appear to impact system performance.

The SPT system was also migrated to an upgraded host computer, transitioning from a Windows x86 to a Windows x64 operating system. This allowed for the implementation code to be run on a more recent MATLAB software version, improving reliability of the

system. In addition, the enhanced computational power likely increased the efficiency of executing the self-pacing control algorithm, though further testing would be required to quantify this and be certain.

A key algorithmic improvement in the updated system was the shift from using only measurements of user position to adjust treadmill speed, to incorporating estimates of both user position and velocity using a Kalman filter, which refined estimates using real-time measurements from the system. While this made the self-pacing algorithm more complex, prior work and open-source implementation code (Song, 2020) was leveraged to facilitate efficient implementation, though several modifications still had to be made to integrate the code with the development goals and constraints in the HAL.

In addition, the updated self-pacing control algorithm eliminated reliance on user leg length, which previously required manual measurement and a hard coded input for each user. Although the updated system still required an individual measurement of user mass, this value was easily obtained computationally by sampling force data from the instrumented treadmill at the very beginning of implementation code execution. Thus, the algorithm's dependence on manually hard-coded parameters was reduced in the updated SPT system.

Another important change to note is that the updated system was designed with future usability in mind. By eliminating reliance on Simulink, a platform not currently used in the HAL, the updated SPT system is more accessible to researchers in the HAL who are more often familiar with standard MATLAB scripts.

Overall, the advancements in the updated SPT system in the HAL significantly improved its usability and accessibility for future studies of walking and running. Additional future testing (discussed in Section 6.1) will further validate its performance and optimize the system for use in future gait research.

5.8 Practical Insights

The evaluation and implementation of the original SPT system in the HAL, along with the design and implementation of the updated SPT system, revealed several practical insights that enhance the comprehensive guidance on SPT system development presented in this thesis. These insights go beyond what is available in existing literature, providing more specific guidance on the development of SPT systems.

One of the most critical takeaways from developing the SPT system in the HAL was the importance of accurately understanding and interpreting sensor measurements. Throughout the development process, a significant portion of time was devoted to acquiring signals from the instrumented treadmill and interpreting their meaning (Secs. 5.3.2 and 5.6.2). In particular, it was critical to understand the units of the force and moment signals and their directional alignment relative to defined sensor origins and positive axis directions. In addition, considerable effort was spent ensuring that sensor noise was properly managed (Sec. 5.5.1). Despite this work being time-intensive, it was both valuable and essential. Any misinterpretation of sensor signals - whether due to incorrect assumptions about signal units or reference frames, system noise, crossover steps or improper data acquisition (e.g. pulling the wrong data channel) - could lead to erroneous system outputs, causing system instability and risking user safety.

Another key takeaway from the hands-on development of the SPT system in the HAL was the high-level of importance that needed to be placed on user safety during system testing, especially early on in system development. When first implementing the control algorithm on system hardware, it was difficult to predict exactly how the treadmill would respond, making initial testing with real users inherently risky. To mitigate the risk, early tests were conducted at lower speeds known to be safe and with restricted treadmill accelerations to prevent overly abrupt system responses. As confidence in system performance grew, maximum velocities and accelerations were gradually increased, but user safety re-

mained a high priority. Users were always instructed to start walking while holding onto the handrails of the treadmill and an emergency stop button was within reach of the user at all times in case they wished to immediately stop the treadmill. Testing was only conducted with young, healthy volunteers, who were capable of stepping or hopping off the treadmill belts if the system responded poorly.

Finally, it was noted that significant value was gained from leveraging existing knowledge of SPT systems, as acquired from the literature, as well as open-source code from existing implementations. Rather than developing an entirely new SPT system, incorporating insights and available resources from previous work on SPT systems significantly improved the efficiency of SPT system development. This approach may be especially beneficial for those developing their first SPT system for gait research and is highly recommended, where feasible, to accelerate the development process.

Chapter 6

Conclusion

6.1 Future Work

With respect to the development of SPT systems for gait research within the larger human movement research community, further investigation is needed to better understand the similarities, differences, advantages and limitations of various self-pacing control algorithms and system hardware. While Chapter 3 provided a broad and high-level overview of existing SPT systems, a more extensive systematic review of existing systems or a meta-analysis of their performance outcomes could offer deeper insights and improve the guidance on system development established in this thesis. In addition, comparative studies that implement multiple self-pacing control algorithms on the same hardware, or the same algorithm on differing hardware, could provide value. Only a few studies have directly compared the performance of different self-pacing control algorithms (Wei et al., 2020; Song et al., 2020). Though comparisons may be challenging when control objectives differ (e.g. evaluating sprinting performance in elite athletes versus providing gait rehabilitation to post-stroke populations) studies aimed at identifying relative advantages of specific algorithms and hardware configurations would provide valuable insights to facilitate future SPT system development.

With respect to the SPT system developed in the HAL, further quantification and testing of system performance should be pursued. For example, Kalman filter estimates of user position and velocity should be validated against ground truth measurements of these variables obtained from the Vicon motion capture system. This assessment would go be-

yond simple verification of whether the estimates follow the user's movement pattern on the treadmill, as was examined in Section 5.6.2 (Fig. 5·13 and Fig. 5·14), instead providing a direct measure of accuracy. In addition, a structured approach to tuning the gains in the self-pacing control law (Eq. 5.23) should be implemented to refine and optimize system performance. Additional algorithmic development of the position-based safety mechanism should be pursued as well. Based on the results of the crossover detection tests (Sec. 5.6.2), the safety mechanism could be improved by making it more robust to erroneous force plate data arising from crossover.

To better understand the user experience of the updated SPT system, a more diverse group of volunteers should be recruited to evaluate the comfort and usability of the system. This assessment could be quantified through the use of questionnaires designed to obtain user feedback. Furthermore, the system's ability to precisely regulate treadmill speed should be tested by instructing volunteers to match predefined target treadmill speeds. To evaluate the system's ability to simulate overground walking, comparative studies of self-selected walking speeds on the SPT versus overground should be conducted. Finally, the operator experience of the system should be evaluated as well, by having different HAL researchers use the system and report their perceptions.

6.2 Final Remarks

The purpose of this thesis was to provide comprehensive guidance on the development of SPT systems for future gait research. By reviewing foundational principles (Ch. 2), examining existing implementations (Ch. 3) and discussing critical design considerations (Ch. 4), a structured framework for SPT system development was established. To go beyond theoretical insights, hands-on evaluation of an SPT system in the HAL at Boston University, along with the design and implementation of an updated system, was completed, providing practical insights that reinforced the established framework (Ch. 5). Ultimately,

by distilling knowledge of SPT systems gained from the literature and documenting both the successes and challenges encountered during real-world SPT system development, this thesis offers a valuable resource for researchers looking to develop their own SPT systems for future gait research.

Appendix A

Code

The following repository contains code and files for the execution of both the original and updated SPT systems in the Human Adaptation Lab at Boston University: GitHub Repository.

References

Alton, F., Baldey, L., Caplan, S., and Morrissey, M. C. (1998). A kinematic comparison of overground and treadmill walking. *Clinical Biomechanics (Bristol, Avon)*, 13(6):434–440.

Bowtell, M. V., Tan, H., and Wilson, A. M. (2009). The consistency of maximum running speed measurements in humans using a feedback-controlled treadmill, and a comparison with maximum attainable speed during overground locomotion. *Journal of Biomechanics*, 42(15):2569–2574.

Cahanin, R., Fallavollita, A., Burley, T., and McQuiston, S. J. (2024). The reliability of clinical tools with and without ultrasound guidance to measure leg-length inequality. *Ultrasound (Leeds, England)*, 32(2):86–93.

Canete, S. and Jacobs, D. A. (2021). Novel velocity estimation for symmetric and asymmetric self-paced treadmill training. *Journal of Neuroengineering and Rehabilitation*, 18(1):27.

Castano, C. R. and Huang, H. J. (2021). Speed-related but not detrended gait variability increases with more sensitive self-paced treadmill controllers at multiple slopes. *PLoS One*, 16(5):e0251229.

Castano, C. R., Lee, L. D., and Huang, H. J. (2023). Speeding up: Discrete mediolateral perturbations increased self-paced walking speed in young and older adults. *Gait & Posture*, 102:198–204.

Cha, M., Han, S., Kim, H., and Mun, D. (2017). User-driven treadmill using walking speed estimated from plantar pressure sensor. *Electronics Letters*, 53(8):524–526.

Choi, J.-S., Kang, D.-W., Seo, J.-W., and Tack, G.-R. (2017). Fractal fluctuations in spatiotemporal variables when walking on a self-paced treadmill. *Journal of Biomechanics*, 65:154–160.

Crenna, F., Rossi, G. B., and Berardengo, M. (2021). Filtering biomechanical signals in movement analysis. *Sensors (Basel, Switzerland)*, 21(4580):1424–8220.

Dingwell, J. B., Cusumano, J. P., Cavanagh, P. R., and Sternad, D. (2001). Local dynamic stability versus kinematic variability of continuous overground and treadmill walking. *Journal of Biomechanical Engineering*, 123(1):27–32.

Dong, H. W., Meng, J., and Luo, Z. (2011). Real-time estimation of human's intended walking speed for treadmill-style locomotion interfaces. In *2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence*, pages 14–19.

Feasel, J., Whitton, M. C., Kassler, L., Brooks, F. P., and Lewek, M. D. (2011). The integrated virtual environment rehabilitation treadmill system. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 19(3):290–297.

Fruet, D., Zignoli, A., Modena, R., Pellegrini, B., Gastaldi, L., and Bortolan, L. (2024). Design and development of a feedback system for automatic treadmill speed adaptation. In *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*.

Fukuchi, C. A., Fukuchi, R. K., and Duarte, M. (2019). Effects of walking speed on gait biomechanics in healthy participants: a systematic review and meta-analysis. *Systematic Reviews*, 8(1):153.

Gembalczyk, G., Duda, S., Switonski, E., and Mezyk, A. (2020). Fuzzy controller for the treadmill speed adaptation system in mechatronic device for gait reeducation. *Journal of Intelligent & Fuzzy Systems*, 39(5):7757–7767.

Gogia, P. P. and Braatz, J. H. (1986). Validity and reliability of leg length measurements. *The Journal of Orthopaedic and Sports Physical Therapy*, 8(4):185–188.

Hunt, K. J., Anandakumaran, P., Loretz, J. A., and Saengsuwan, J. (2018). A new method for self-paced peak performance testing on a treadmill. *Clinical Physiology and Functional Imaging*, 38(1):108–117.

Ibala, E., Coupaud, S., and Kerr, A. (2019). Comparison of the muscle pattern variability during treadmill walking (fixed and self-pace) and over ground walking of able-bodied adults. *Journal of Annals of Bioengineering*, 2019(1):1–11.

Kao, P.-C. and Pierro, M. A. (2021). Dual-task treadmill walking at self-paced versus fixed speeds. *Gait & Posture*, 89:92–101.

Kim, J., Gravunder, A., and Park, H.-S. (2015). Commercial motion sensor based low-cost and convenient interactive treadmill. *Sensors*, 15(9):23667–23683.

Kim, J., Stanley, C. J., Curatalo, L. A., and Park, H.-S. (2012). A user-driven treadmill control scheme for simulating overground locomotion. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual International Conference*, 2012:3061–3064.

Kimel-Naor, S., Gottlieb, A., and Plotnik, M. (2017). The effect of uphill and downhill walking on gait parameters: A self-paced treadmill study. *Journal of Biomechanics*, 60:142–149.

Lacey, T. and Thacker, N. (1998). Tutorial: The kalman filter. Technical report, University of Manchester, Imaging Science and Biomedical Engineering Division.

Lee, S. J. and Hidler, J. (2008). Biomechanics of overground vs. treadmill walking in healthy individuals. *Journal of Applied Physiology (Bethesda, Md.: 1985)*, 104(3):747–755.

Lewis, C. L., Halverstadt, A. L., Graber, K. A., Perkins, Z., Keiser, E., Belcher, H., Khuu, A., and Loverro, K. L. (2021). Individuals with pre-arthritic hip pain walk with hip motion alterations common in individuals with hip oa. *Frontiers in Sports and Active Living*, 3:719097.

Lichtenstein, L., Barabas, J., Woods, R. L., and Peli, E. (2007). A feedback-controlled interface for treadmill locomotion in virtual environments. *ACM Transactions on Applied Perception*, 4(1):7.

Malatesta, D., Canepa, M., and Fernandez, A. M. (2017). The effect of treadmill and overground walking on preferred walking speed and gait kinematics in healthy, physically active older adults. *European Journal of Applied Physiology*, 117(9):1833–1843.

Minetti, A. E., Boldrini, L., Brusamolin, L., Zamparo, P., and McKee, T. (2003). A feedback-controlled treadmill (treadmill-on-demand) and the spontaneous speed of walking and running in humans. *Journal of Applied Physiology (Bethesda, Md.: 1985)*, 95(2):838–843.

Parvataneni, K., Ploeg, L., Olney, S. J., and Brouwer, B. (2009). Kinematic, kinetic and metabolic parameters of treadmill versus overground walking in healthy older adults. *Clinical Biomechanics (Bristol, Avon)*, 24(1):95–100.

Pei, Y., Biswas, S., Fussell, D. S., and Pingali, K. (2019). An elementary introduction to kalman filtering. *Communications of the ACM*, 62(11):122–133.

Pirker, W. and Katzenschlager, R. (2017). Gait disorders in adults and the elderly : A clinical guide. *Wiener Klinische Wochenschrift*, 129(3-4):81–95.

Plotnik, M., Azrad, T., Bondi, M., Bahat, Y., Gimmon, Y., Zeilig, G., Inzelberg, R., and Siev-Ner, I. (2015). Self-selected gait speed - over ground versus self-paced treadmill walking, a solution for a paradox. *Journal of NeuroEngineering and Rehabilitation*, 12(1):20.

Rao, R. P., Sara, L. K., Perkins, Z. E., Dwyer, M. K., and Lewis, C. L. (2024). Females with hip pain walk with altered kinematics at peaks and throughout the gait cycle. *Clinical Biomechanics (Bristol, Avon)*, 118:106314.

Ray, N. T., Knarr, B. A., and Higginson, J. S. (2018). Walking speed changes in response to novel user-driven treadmill control. *Journal of Biomechanics*, 78:143–149.

Reneaud, N., Gerus, P., Guerin, O., Garda, M., Piche, E., Chorin, F., and Zory, R. (2022). 6mwt on a new self-paced treadmill system compared with overground. *Gait & Posture*, 92:8–14.

Riley, P. O., Paolini, G., Croce, U. D., Paylo, K. W., and Kerrigan, D. C. (2007). A kinematic and kinetic comparison of overground and treadmill walking in healthy subjects. *Gait & Posture*, 26(1):17–24.

Scheadler, C. M. and Devor, S. T. (2015). Vo2max measured with a self-selected work rate protocol on an automated treadmill. *Medicine & Science in Sports & Exercise*, 47(10):2158–65.

Schmitt, A. C., Baudendistel, S. T., Lipat, A. L., White, T. A., Raffegeau, T. E., and Hass, C. J. (2021). Walking indoors, outdoors, and on a treadmill: Gait differences in healthy young and older adults. *Gait & Posture*, 90:468–474.

Sloot, L. H., van der Krogt, M. M., and Harlaar, J. (2014). Self-paced versus fixed speed treadmill walking. *Gait & Posture*, 39(1):478–484.

Song, S. (2020). Self-paced-treadmill repository. https://github.com/smsong/self-paced-treadmill/tree/JNER2020.

Song, S., Choi, H., and Collins, S. H. (2020). Using force data to self-pace an instrumented treadmill and measure self-selected walking speed. *Journal of Neuroengineering and Rehabilitation*, 17(1):68–5.

Souman, J. L., Giordano, P. R., Frissen, I., Luca, A. D., and Ernst, M. O. (2010). Making virtual walking real: Perceptual evaluation of a new treadmill control algorithm. *ACM Transactions on Applied Perception*, 7(2):1–14.

Theunissen, K., Hooren, B. V., Plasqui, G., and Meijer, K. (2022). Self-paced and fixed speed treadmill walking yield similar energetics and biomechanics across different speeds. *Gait & Posture*, 92:2–7.

Truzman, S., Revach, G., Shlezinger, N., and Klein, I. (2024). Outlier-insensitive kalman filtering: Theory and applications. *IEEE Sensors Journal*.

van der Krogt, M. M., Sloot, L. H., and Harlaar, J. (2014). Overground versus self-paced treadmill walking in a virtual environment in children with cerebral palsy. *Gait & Posture*, 40(4):587–593.

van Ingen Schenau, G. J. (1980). Some fundamental aspects of the biomechanics of overground versus treadmill locomotion. *Medicine and Science in Sports and Exercise*, 12(4):257–261.

von Zitzewitz, J., Bernhardt, M., and Riener, R. (2007). A novel method for automatic treadmill speed adaptation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 15(3):401–409.

Wang, H. and Hunt, K. J. (2023). Self-paced heart rate control for treadmill exercise. *Frontiers in Control Engineering*, 4.

Wang, W., Yang, K., Zhu, Y., and Mu, H. (2020). Speed adaptation and acceleration ripple suppression of treadmill user system using a virtual force moment balance model. *Transactions of the Institute of Measurement and Control*, 42(2):322–329.

Wei, W., Kaiming, Y., Yu, Z., Yuyang, Q., and Chenhui, W. (2020). A comparison of variability and gait dynamics in spatiotemporal variables between different self-paced treadmill control modes. *Journal of Biomechanics*, 110:109979.

Whittle, M. W. (2007). *Chapter 2 - Normal gait*, pages 47–100. Butterworth-Heinemann, Edinburgh.

Wiens, C., Denton, W., Schieber, M. N., Hartley, R., Marmelat, V., Myers, S. A., and Yentes, J. M. (2019). Walking speed and spatiotemporal step mean measures are reliable during feedback-controlled treadmill walking; however, spatiotemporal step variability is not reliable. *Journal of Biomechanics*, 83:221–226.

Yoon, J., Park, H.-S., and Damiano, D. L. (2012). A novel walking speed estimation scheme and its application to treadmill control for gait rehabilitation. *Journal of Neuroengineering and Rehabilitation*, 9:62–62.

CURRICULUM VITAE

Zoe Perkins
perkinsz@bu.edu

EDUCATION

Boston University, Boston, MA
M.S. Mechanical Engineering
May 2025

Boston University, Boston, MA
B.A. Biochemistry and Molecular Biology
May 2021

RESEARCH EXPERIENCE

Human Adaptation Lab, Boston University, Boston, MA
Lab Manager
Oct 2021 - Sep 2024

Human Adaptation Lab, Boston University, Boston, MA
Undergraduate Student Researcher
Apr 2019 - May 2021

PUBLICATIONS

1. Rao R, Sara LK, **Perkins Z**, Dwyer M, Lewis CL. Altered gait kinematics in females with hip pain exist at peaks and throughout the gait cycle. *Clinical Biomechanics (Bristol, Avon)*. 2024; 118:106314. DOI: 10.1016/j.clinbiomech.2024.106314

2. Lewis CL, Halverstadt AL, Graber KA, **Perkins Z**, Keiser E, Belcher H, Khuu A, Loverro KL. Individuals With Pre-arthritic Hip Pain Walk With Hip Motion Alterations Common in Individuals With Hip OA. *Frontiers in Sports and Active Living*. 2021; 3:719097. DOI: 10.3389/fspor.2021.719097

AWARDS

- Boston University College of Engineering Graduate Scholarship, AY 23-24, AY 24-25
- Boston University Undergraduate Research Opportunities Award, Summer 2020