Ascending New Heights: Enhancing Bipedal Robotic Locomotion through Stair Climbing

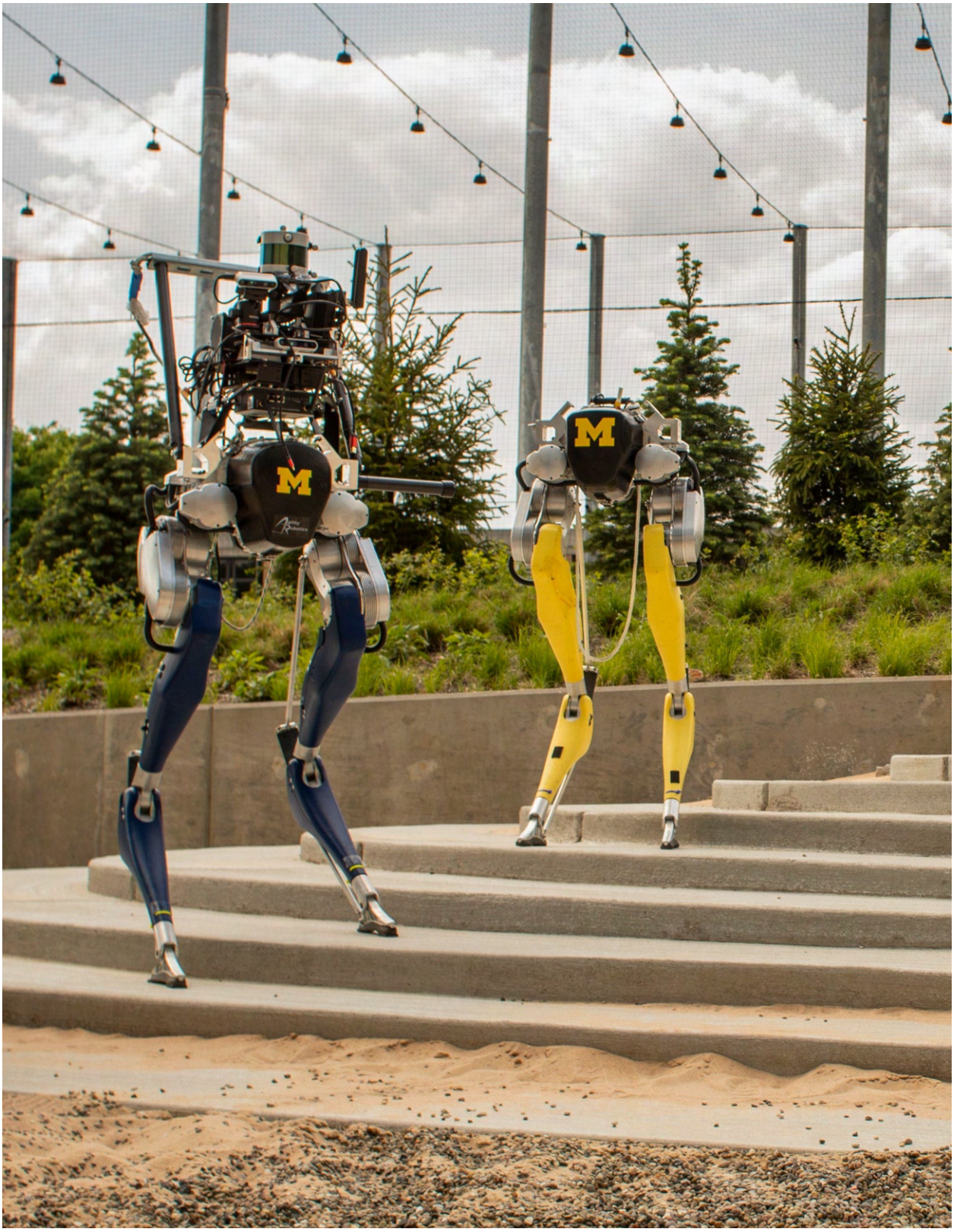
by

Oluwami Dosunmu-Ogunbi

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in the University of Michigan
2024

Doctoral Committee:

Professor Jessy W Grizzle, Chair
Professor Kira Barton
Professor Chad Jenkins
Associate Professor Ramanarayan Vasudevan



Oluwami Dosunmu-Ogunbi

wami@umich.edu

ORCID iD: 0000-0002-9585-8119

© Oluwami Dosunmu-Ogunbi 2024

For all the little Black girls who were told that they could not climb...let them witness your ascension.

Acknowledgments

My successful 3.5-year journey to become the first Black woman to earn a PhD in Robotics at the University of Michigan was shaped by many supportive individuals.

Thank you to my committee members. Eternal gratitude to my advisor, Professor Jessy Grizzle, who, despite being ready for retirement after a long and illustrious career, moved mountains to accommodate me as his last PhD student. Your commitment afforded me an invaluable opportunity to study at the highest level, and for that, I am profoundly thankful. Special recognition to Professor Chadwicke Jenkins, my mentor, who intervened swiftly during a challenging moment in my academic career, facilitating my transition to the Robotics PhD program. To Professor Ramanarayan Vasudevan for opening doors as I applied for jobs and for allowing me to use his Cassie Maize robot, crucial for the hardware experiments in this thesis. To Professor Kira Barton, your invaluable feedback on job applications and support during qualifying exams were crucial, putting me at ease and drawing out my potential.

Thank you to Michigan Robotics staff and faculty. To Professor Robert Gregg for support during qualifying exams and for professional development. To Alyssa Emigh and Professor Peter Gaskell for supporting the Cassie robot, ensuring continuity in hardware experiments. To Samantha, Jamie, Denise, Kimberly, and countless others, thank you for your support.

To my lab mentor, Yukai, and senior lab mates—Omar, Eva, Grant, Bruce, Ray, and Lu—thank you for fostering a supportive environment during my challenging start in the midst of the COVID-19 pandemic. To my outstanding Masters and Undergraduate mentees, and in particular, Aayushi Shrivastava, whose exceptional contributions played a pivotal role in shaping much of this thesis. It has truly been an honor working alongside you.

Thank you to my family. To my parents, Doyin and Sesi, my first and loudest cheerleaders, your unconditional love guided me through many challenges. To my siblings Olu and Oyin, your consistent praise not only made me proud of what I do as an engineer and roboticist but also provided unwavering strength. To my extended family for their love and support.

To my friends and colleagues in Michigan, Illinois, Alabama, and around the world, your companionship through deadlines, exams, concerts, book clubs, travel, and identity crises, as well as other precious shared experiences, will forever hold a special place in my heart. Your collective support has been a beacon of strength throughout my academic journey.

"The Waminator" thrives because of all of you.

TABLE OF CONTENTS

List of Figures

FIGURE

List of Appendices

List of Abbreviations

2D 2-Dimensional

3D 3-Dimensional

ALIP Angular Momentum Linear Inverted Pendulum

A-SLIP Angular Momentum Spring-Loaded Inverted Pendulum

CoM center of mass

DoF degrees-of-freedom

FROST Fast Robot Optimization and Simulation Toolkit

IMU Inertial Measurement Unit

LIP Linear Inverted Pendulum

LQR Linear Quadratic Regulator

MPC Model Predictive Control

ODE Ordinary Differential Equation

PBC Passivity-Based Control

PID Proportional-Integral-Derivative

QP Quadratic Program

URDF Unified Robotics Description Format

Abstract

A new control paradigm using angular momentum and foot placement as state variables in the linear inverted pendulum model has expanded the realm of possibilities for the control of bipedal robots. This new paradigm, known as the Angular Linear Inverted Pendulum (ALIP) model, has shown effectiveness in cases where a robot's center of mass height can be assumed to be constant or near constant as well as in cases where there are no non-kinematic restrictions on foot placement. Walking up and down stairs violates both of these assumptions, where center of mass height varies significantly within a step and the geometry of the stairs restrict the effectiveness of foot placement.

In this thesis, we explore a variation of the ALIP model that allows the length of the virtual pendulum formed by the robot's stance foot and center of mass to follow smooth trajectories during a step. We couple this model with a control strategy constructed from a novel combination of virtual constraint-based control and a model predictive control algorithm to stabilize a stair-climbing gait that does not solely rely on foot placement. Simulations on a 20-degree of freedom model of the Cassie biped in the SimMechanics simulation environment show that the controller is able to achieve periodic gait. Hardware experiments also show promise of improving the robustness of walking gaits on inclined surfaces.

Chapter 1

Introduction

1.1 Motivation

In a world where safety and efficiency are paramount, the integration of robots into various domains has become increasingly prevalent [7, 8]. From disaster relief efforts [9, 10, 11] to industrial automation [12, 13, 14, 15], underwater operations [16], and healthcare care [17], robots have the potential to alleviate risks and enhance the overall quality of life for humans. While many significant advancements in legged robotics have been achieved over the past decades, there are still crucial challenges that must be addressed to unlock the true potential of bipedal robots [18, 19, 20, 21, 22].

One fundamental aspect of human-centric environments is the presence of stairs—a ubiquitous feature that poses a significant obstacle for both humans and bipedal robots [23, 24, 25, 26]. Conquering this challenge requires innovative approaches to robot control system design, specifically tailored for bipedal machines. This thesis aims to present a novel solution by proposing a method for asymptotically stable stair climbing that addresses the unique characteristics and constraints of underactuated bipedal robots.

Drawing inspiration from the inverted pendulum model [27, 28], which is widely used to analyze bipedal locomotion, our research introduces a novel variation that allows for dynamic variations in the robot's center of mass height within each

step. By combining virtual constraint-based control and Model Predictive Control, we achieve a stable stair climbing gait that enables bipedal robots to ascend stairs efficiently and safely.

Throughout this thesis, we will delve into the theoretical foundations of our proposed method, provide a comprehensive analysis of the control algorithms employed, and present experimental results to validate the effectiveness and feasibility of our approach. By shedding light on the challenges faced by bipedal robots and offering a promising solution for stair climbing, we aim to contribute to the advancement of robotics and pave the way for a more robot-assisted world.

1.2 Objectives

The overarching goal of this thesis is to develop a gait design method for stair climbing specifically tailored for underactuated bipedal robots. To achieve this goal, the following specific objectives will be pursued:

1. Conduct a comprehensive review of existing literature and research on bipedal locomotion, with a particular focus on stair-climbing techniques and challenges faced by bipedal robots in traversing stairs.

2. Propose a novel variation of the inverted pendulum model that allows for variations in the center of mass height within a step, enabling bipedal robots to adapt their gait for stair climbing.

3. Develop virtual constraints to facilitate stability and precise control of the bipedal robot during stair climbing using the proposed variation of the inverted pendulum model.

4. Implement a Model Predictive Control-based ankle actuation strategy that works with the virtual constraints to stabilize the stair-climbing gait.

5. Evaluate the proposed stair-climbing method through extensive simulations to assess its stability, efficiency, and adaptability to different stair configurations.

6. Conduct experimental validations using an underactuated bipedal robot to assess the practical feasibility and performance of the proposed method in real-world scenarios.

7. Analyze and compare the results obtained from simulations and experiments, identifying the strengths, limitations, and areas for improvement of the proposed stair-climbing method.

8. Provide recommendations for future research and development in the field of bipedal robot mobility, particularly in the context of stair climbing, aiming to inspire further advancements and innovations.

By accomplishing these objectives, this thesis aims to contribute to the field of robotics by presenting a practical and effective solution for stair climbing that enhances the mobility and versatility of underactuated bipedal robots.

1.3 Thesis Overview

In the following chapters, we will introduce the bipedal robot platform used in this thesis as well as modeling methodology (Chapter 2), explore the existing literature on bipedal locomotion and discuss the technical details of our proposed method (Chapter 3), present simulation and experimental results (Chapters 4 and 5), and finally, discuss the implications and potential future developments that can arise from our research (Chapter 6). Through this exploration, we aspire to foster innovation and inspire further advancements in the field of robotics, ultimately enabling the creation of more capable and versatile robots that can operate effectively in human-centric environments.

Chapter 2

Laying the Blueprint - Modeling the Cassie Biped

Before introducing our novel stair-climbing algorithm, a comprehensive acquaintance with the selected robot platform for testing and validation is essential. We thus begin with introducing the robot that we use for this work. This chapter explores the mathematical modeling of the Cassie Biped robot. By developing a comprehensive model that captures its mechanical structure and dynamics, we gain valuable insights into its stability and locomotion patterns, providing a scaffold on which to apply our stair-climbing algorithm.

2.1 The Cassie Bipedal Robot

The stair-climbing algorithm presented in this dissertation is validated on the Cassie robot. Cassie (shown in Figure 2.1) is a bipedal robot developed by the company Agility Robotics [29]. Standing at 1.5 meters tall and weighing 35 kilograms (4.92 feet and 77 pounds), its design is inspired by the structure and movements of a cassowary, an ostrich-like bird. Only twelve Cassie robots were built, nine of which were purchased by research labs across the United States (University of Michigan, Georgia Institute of Technology, California Institute of Technology, University of Pennsylvania, Oregon State, University of California Berkeley, Florida A&M University - Florida State University, and Harvard), while the remaining three remained in possession of Agility Robotics. The University of Michigan owns two, Cassie Blue and Cassie Maize, which are identical, save for a few features (e.g., different spring

stiffnesses). The first Cassie was released in 2017, but as of February 2023, Agility Robotics no longer supports the Cassie platform. This poses a logistical challenge for the hardware experiments in this thesis, as hardware experimentation began after Cassie's warranty ended.

Cassie has ten motors and 20 degrees-of-freedom (DoF), making it an under-actuated robot. One of the key features researchers implement on Cassie is the ability to autonomously maintain stability and recover from perturbations. Using a combination of an Inertial Measurement Unit (IMU) and encoders in each of its fourteen joints (labeled in Figure 2.2 and Figure 2.3), Cassie is able to constantly monitor its position, orientation, and ground contact. This sensory information is processed by sophisticated onboard algorithms to generate precise motor commands and maintain balance [30, 31, 32, 33, 34, 35].

Cassie is capable of walking on various terrains, including flat surfaces [1], uneven ground [35, 36], gentle slopes [31], and stairs [37, 38, 39]. The applications of Cassie are wide-ranging. It has been primarily used in research and development for studying bipedal locomotion, gait analysis, and balance control. Additionally, Cassie has the potential to be used in various practical applications, such as package delivery, search and rescue operations, and industrial tasks that require navigating human environments. Moreover, controllers built for the Cassie robot can be applied to other bipedal robot platforms [39, 40].

Overall, Cassie represents a significant advancement in bipedal robotics, combining sophisticated control algorithms, sensors, and a biomechanical design to achieve stable and natural walking capabilities. Its versatility and potential applications make it a promising platform for future advancements in robotics and automation.

Now that we have a robot platform, we need to understand how to mathematically model it and the various forces acting on it to be able to effectively apply our novel stair-climbing algorithm.

2.2 The Robot Equations

The equations of motion, also known as the **robot equations**, describe the relationship between the forces acting on a physical object (e.g., robot) and its resulting motion [41, 42, 43, 44]. There are several approaches to generating a robot's equations of motion, including Newtonian mechanics (which uses Newton's Laws) and the Lagrangian (which uses energy). The method one chooses to use depends on variety of practical and strategic factors.

This thesis chooses to use the Lagrangian approach for the following reasons:

- Generalization and Flexibility: The Lagrangian approach provides a generalized framework that can handle a wide range of physical systems, including those with constraints, non-conservative forces, and generalized coordinates. It offers a unified treatment for different types of systems, making it applicable to a broader class of problems.

- Simplicity and Clarity: The Lagrangian formulation often simplifies the mathematical analysis compared to other approaches. It avoids the need to consider individual forces acting on a system and focuses instead on the system's overall energy and the variation of its generalized coordinates. This can make the derivation of equations of motion more straightforward, especially for complex systems.

- Coordinate Independence: The Lagrangian approach is coordinate-independent, allowing for a more elegant treatment of systems with generalized coordinates that might not be easily expressed in Cartesian coordinates. This makes it particularly useful for describing systems with non-linear or non-Cartesian geometries (like Cassie).

- Conservation Laws: The Lagrangian formulation naturally incorporates the conser-

vation laws, such as the conservation of energy and momentum, as consequences of symmetries in the system. These conservation laws can provide valuable insights into the system's behavior without explicitly considering all the forces acting on the system.

Deriving the robot equations is a non-trivial task that can take dozens to hundreds of pages to write, especially for robots as complex as the Cassie biped. Thus, we resort to software such as Matlab and Mathamatica to handle these derivations [45]. Nevertheless, it is useful to understand the basics behind the math that the software employs. Appendix A walks through the derivation of the robot equations using Lagrangian mechanics on a robot approximated as point masses. Note that we do not use the point mass approximation on Cassie to generate the robot equations in our application in this thesis. Instead, we use distributed mass, relying on a Unified Robotics Description Format (URDF) of Cassie to capture Cassie's structure and kinematics. However, understanding the Lagrangian mechanics deviation of the robot equations on a point-mass model provides useful insights on how the derivation would apply on a distributed mass system.

2.2.1 Cassie's Robot Equations

Bipedal locomotion, such as with stair climbing, can be best characterized using a *hybrid system*–a system that displays both continuous and discrete behavior [46, 2]. The continuous phase describes the dynamics of one foot supporting the robot and the other swinging forward, while the discrete phase describes the transitions between left and right feet. The "stance leg" is defined as the leg that is planted on the ground during walking motion. Conversely, the "swing leg" refers to the leg whose foot is progressing forward.

Using Lagrangian mechanics, one obtains a second-order differential equation to

describe the continuous dynamics for the Cassie biped:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = J_{st}^T F + J_s^T F_s + Bu \tag{2.1}$$

where $D \in \mathbb{R}^{20\times 20}$ is the mass inertial matrix, $C \in \mathbb{R}^{20\times 20}$ is the centrifugal and coriolis forces matrix, $G \in \mathbb{R}^{20\times 1}$ is the gravitational vector, $J_{st} \in \mathbb{R}^{5\times 20}$ is the stance foot Jacobian (we assume that the blade foot has two points of contact), $F \in \mathbb{R}^{5\times 1}$ is the ground reaction force acting on the stance foot, $J_s \in \mathbb{R}^{4\times 20}$ is the Jacobian of the springs, $F_s \in \mathbb{R}^{4\times 1}$ are the forces acting from the springs, $B \in \mathbb{R}^{20\times 10}$ is the input matrix, $u \in \mathbb{R}^{10\times 1}$ is the motor torque vector, and $q \in \mathbb{R}^{20\times 1}$ is the generalized coordinate vector.

The generalized coordinates of the Cassie robot are defined as follows:

$$q = \begin{bmatrix} q_{\text{floating base}} \\ \\ q_{\text{body}} \end{bmatrix} \tag{2.2}$$

where

$$q_{\text{floating base}} = \left[q_{\text{x}}^{\text{absolute}}, q_{\text{y}}^{\text{absolute}}, q_{\text{z}}^{\text{absolute}}, q_{\text{yaw}}^{\text{absolute}}, q_{\text{pitch}}^{\text{absolute}}, q_{\text{roll}}^{\text{absolute}} \right]^\top \tag{2.3}$$

are the floating base coordinates that represent the pose of the base link with respect to a fixed inertial frame, and

$$\begin{aligned} q_{\text{body}} = \Big[& q_{\text{hip roll}}^{\text{left}}, q_{\text{hip yaw}}^{\text{left}}, q_{\text{hip pitch}}^{\text{left}}, q_{\text{knee}}^{\text{left}}, q_{\text{knee spring}}^{\text{left}}, q_{\text{ankle}}^{\text{left}}, q_{\text{toe}}^{\text{left}}, \\ & q_{\text{hip roll}}^{\text{right}}, q_{\text{hip yaw}}^{\text{right}}, q_{\text{hip pitch}}^{\text{right}}, q_{\text{knee}}^{\text{right}}, q_{\text{knee spring}}^{\text{right}}, q_{\text{ankle}}^{\text{right}}, q_{\text{toe}}^{\text{right}} \Big]^\top. \end{aligned} \tag{2.4}$$

For reasons discussed in the next chapter, we reformulate the equations of motion defined in Equation (2.1) such that the stance ankle torque term is isolated from

the rest of the input terms. Thus,

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = J_{st}^T F + J_s^T F_s + B_1 u_1 + B_9 u_9 \tag{2.5}$$

where $B_9 \in \mathbb{R}^{20\times 9}$ and $u_9 \in \mathbb{R}^{9\times 1}$ are the input matrix and control vector without the stance ankle terms, respectively, and $B_1 \in \mathbb{R}^{20\times 1}$ and $u_1 \in \mathbb{R}^{1\times 1}$ correspond to the column in the input matrix and value in the control vector relating to the stance ankle torque, respectively.

2.2.2 Constraints due to Springs and Contacts

When using a floating based model, the Cassie robot has 20 DoF. The four springs—two on each leg—are very stiff. We therefore approximate them as four holonomic constraints. When Cassie has one foot on the ground—which is the case when walking or climbing stairs—that foot adds another 5 holonomic constraints to the system: constraining the Cartesian position of the front and back of the foot sets the Cartesian position of the foot, as well as its yaw angle and pitch angle. Because the foot is narrow, it is modeled as a blade, allowing roll about the x-axis.

2.3 Building the Virtual Twin - Linear Inverted Pendulum-Based Biped Robot Modeling

There is a common saying coined by the British statistician George E. P. Box that goes "all models are wrong, but some are useful." In the context of bipedal robotics, roboticists have used a range of models to achieve agile movement in their robots [47, 48, 49, 50, 51, 52]. Full-order dynamical models have proven to be too computationally expensive for practical online control calculations and/or it has proven hard to transfer among different robots of similar morphology [53, 54]. More granular models make it easier to apply a variety of control schemes and

perform real-time computations, however, they can also be ineffective in capturing the dominant dynamics of a robot, thus limiting the agility of the closed-loop system (robot plus the controller) [55, 56, 57]. In addition, the sim-to-real gap can be hard to manage [58, 59]. We use a model in this dissertation to more efficiently apply our control scheme that would otherwise prove to be too computational expensive to implement on a full order-model of Cassie in real-time.

2.3.1 The Linear Inverted Pendulum (LIP) Model

The Linear Inverted Pendulum (LIP) model (shown in Figure 2.4) is a simplified representation of the dynamics of a pendulum in an inverted position. It is often used in control theory and robotics to study the stability and control of systems that resemble an inverted pendulum. The LIP model is particularly relevant in the context of balancing systems like those found in applications such as bipedal robots, where the LIP model assumes a point mass fixed on massless legs [27, 28].

The basic idea is to model the dynamics of an inverted pendulum as a linear system around an equilibrium point. The equilibrium point is the point around which the system is linearized. For an inverted pendulum, the equilibrium point is often chosen to be the upright position.

The LIP model is a valuable tool for analyzing the stability and designing control strategies for systems that involve balancing an inverted pendulum. Control methods such as Linear Quadratic Regulator (LQR) or Proportional-Integral-Derivative (PID) controllers are commonly applied to stabilize and control systems modeled using the LIP framework.

Approaches that use the LIP model in bipedal locomotion typically assume a constant center of mass height and use CoM velocity as a means to quantify "balance" (e.g., speed stabilization). These assumptions fail to effectively capture impacts associated with gaits where the CoM height undergoes significant variation

[60].

Recent research shows that angular momentum about the contact point of the stance foot has higher fidelity when applied to realistic robots [61, 60]. This newer paradigm, called the Angular Momentum Linear Inverted Pendulum (ALIP) model, has been used in control strategies to determine foot placement. Critically, angular momentum about the support foot has relative degree three with respect to all motor torques except the stance ankle, where it has relative degree one. Consequently, angular momentum about the support foot is directly controllable via ankle torque and only weakly affected by distal motor torques throughout a step. Furthermore, the transfer of angular momentum property at impact shows that angular momentum about a given contact point is invariant to the impulsive force generated at the contact point [48].

The ALIP model is a reparameterization of the LIP model where the linear velocity of the CoM is replaced by the angular momentum about the contact point as a key variable to "summarize" the state of a robot. For robot models consisting of a single point mass suspended on massless legs, the ALIP model is equivalent to the LIP model. For real robots, with links having distributed mass, reference [61] shows that the ALIP model is superior for making predictions about future state values.

While the ALIP model has proven to be an effective means of achieving agile locomotion over flat ground [61], the model has not yet been demonstrated on tasks that involve rapid changes to CoM height such as stair climbing or climbing onto or off objects. Truly agile bipedal robots must be fitted with a controller that is able to handle rapid changes to CoM height to make them capable of navigating cluttered environments. This work further capitalizes on the benefits of the ALIP model by coupling the angular momentum parameter with CoM angle, a variable that better captures the non-constant behavior of a robot's center of mass in gaits

on uneven surfaces. We discuss this new ALIP model further in the following subsections.

2.3.2 Derivation of the new Angular Momentum Linear Inverted Pendulum (ALIP) Model

The derivation of the new ALIP model is as follows. Assume an inverted pendulum as shown in Figure 2.4, where (x_c, z_c) are the Cartesian position of the CoM with respect to the stance foot. It follows that the angle of the CoM with respect to the stance foot is

$$\theta_c = \arctan\Big(\frac{x_c}{z_c}\Big). \tag{2.6}$$

Taking the derivative with respect to time yields

$$\begin{aligned}\dot{\theta}_c &= \frac{1}{1+(\frac{x_c}{z_c})^2}\Big(\frac{\dot{x}_c z_c - \dot{z}_c x_c}{z_c^2}\Big) \\ &= \frac{1}{z_c^2 + x_c^2}(\dot{x}_c z_c - \dot{z}_c x_c) \\ &= \frac{1}{r_c^2}(\dot{x}_c z_c - \dot{z}_c x_c)\end{aligned} \tag{2.7}$$

where $r_c = \sqrt{x_c^2 + z_c^2}$ is the length of the pendulum. For later use, we rewrite Equation (2.7) as

$$\dot{\theta}_c = \frac{1}{m r_c^2}(m\dot{x}_c z_c - m\dot{z}_c x_c) \tag{2.8}$$

where m denotes total mass.

Given the angular momentum about the contact point L and the angular momentum about the CoM, L_c, the *angular momentum transfer formula* [31] gives

$$L - L_c = m \begin{bmatrix} x_c \\ z_c \end{bmatrix} \wedge \begin{bmatrix} \dot{x}_c \\ \dot{z}_c \end{bmatrix} = m z_c \dot{x}_c - m x_c \dot{z}_c \tag{2.9}$$

where

$$\begin{bmatrix} x_c \\ z_c \end{bmatrix} \wedge \begin{bmatrix} \dot{x}_c \\ \dot{z}_c \end{bmatrix} := \left(\begin{bmatrix} x_c \\ 0 \\ z_c \end{bmatrix} \times \begin{bmatrix} \dot{x}_c \\ 0 \\ \dot{z}_c \end{bmatrix} \right) \bullet \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

Using Equation (2.9), Equation (2.8) becomes

$$\dot{\theta}_c = \frac{L - L_c}{m r_c^2}. \tag{2.10}$$

To complete the model, the time derivative of L, the angular momentum about the stance leg is

$$\begin{aligned} \dot{L} &= m g x_c + \tau \\ &= m g r_c \sin(\theta_c) + \tau, \end{aligned} \tag{2.11}$$

where τ is the torque about the contact point, which we will call stance ankle torque. Note that τ here is equivalent to u_1 in Equation (2.5). Combining Equation (2.10) and Equation (2.11), the dynamical model becomes

$$\begin{aligned} \dot{\theta}_c &= \frac{L - L_c}{m r_c^2} \\ \dot{L} &= m g r_c \sin(\theta_c) + \tau. \end{aligned} \tag{2.12}$$

In [31], it is shown that L_c can be neglected for Cassie-like robots. For the nominal stair-climbing trajectory, $-0.21 \leq \theta_c \leq 0.13$ radians, and hence we can make the approximation $\sin(\theta_c) \approx \theta_c$. This results in the linear time-varying model

$$\begin{aligned} \dot{\theta}_c &= \frac{L}{m r_c^2(t)} \\ \dot{L} &= m g r_c \theta_c + \tau, \end{aligned} \tag{2.13}$$

which we refer to as the ALIP. The model is time-varying because we will assume that $r_c(t)$ evolves according to the nominal periodic orbit.

2.3.3 Remarks on the ALIP Model

When the CoM is controlled to a constant height, the ALIP model becomes linear and time-invariant, and hence admits a closed-form solution. When walking on level ground, a constant CoM assumption renders the impact map linear in the planned horizontal swing foot position.

Walking on stairs violates two of the key assumptions made above: a) the CoM height of the robot must vary to pass from one step to the next, and b) the run of each step of the stair severely restricts horizontal foot placement, effectively eliminating it as a control decision variable. This new version of the ALIP model from [31] facilitates accounting for varying pendulum length. We also introduced stance-leg ankle torque into the model so that it can be used as a control variable. This is discussed further in the next chapter.

2.4 Impact Map

An *impact map* [2, 62] is a means of capturing a system's state after impact (i.e., when the swing foot touches the floor in a walking trajectory). Specifically, the impact map calculates what the generalized velocities and forces acting on the system are following impact by integrating the dynamics over an instantaneous period of time. This period is bounded from t^- (time before impact) and t^+ (time after impact). Because impact occurs when the swing foot touches the ground, the swing foot is also included in the dynamics. Thus, Equation (2.5) extends to become

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = J_{st}^T F + J_s^T F_s + B_1 u_1 + B_9 u_9 + J_{sw}^T \delta F_{sw} \tag{2.14}$$

where $J_{sw} \in \mathbb{R}^{5\times 20}$ is the swing foot Jacobian and $\delta F_{sw} \in \mathbb{R}^{4\times 1}$ are the impulsive ground reaction forces acting on the swing foot acting at impact. Integrating 2.14

from time before impact t^- to time after impact t^+ yields

$$D(q)(\dot{q}^+ - \dot{q}^-) = J_{s,sw}^T F_{s,sw} \tag{2.15}$$

where $\dot{q}^- \in \mathbb{R}^{20\times 1}$ are the generalized velocities before impact and $\dot{q}^+ \in \mathbb{R}^{20\times 1}$ are the generalized velocities after impact, and $F_{s,sw} \in \mathbb{R}^{9\times 1}$ are the impulsive forces acting on the springs and swing foot, and $J_{s,sw} \in \mathbb{R}^{9\times 20}$ is the spring and swing foot Jacobian.

We assume a non-slip constraint. That is, swing foot velocity is zero after impact:

$$J_{sw}\dot{q}^+ = 0. \tag{2.16}$$

Combining Equation (2.15) and Equation (2.16) yields,

$$\begin{bmatrix} \dot{q}^+ \\ F_{s,sw} \end{bmatrix} = \begin{bmatrix} D & -J_{s,sw}^T \\ J_{s,sw} & 0 \end{bmatrix}^{-1} \begin{bmatrix} D\dot{q}^- \\ 0 \end{bmatrix}. \tag{2.17}$$

Equation (2.17) is the impact map for the Cassie robot.

Figure 2.1: The Cassie Blue bipedal robot built by Agility Robotics roaming through a fiery field at the University of Michigan [1].

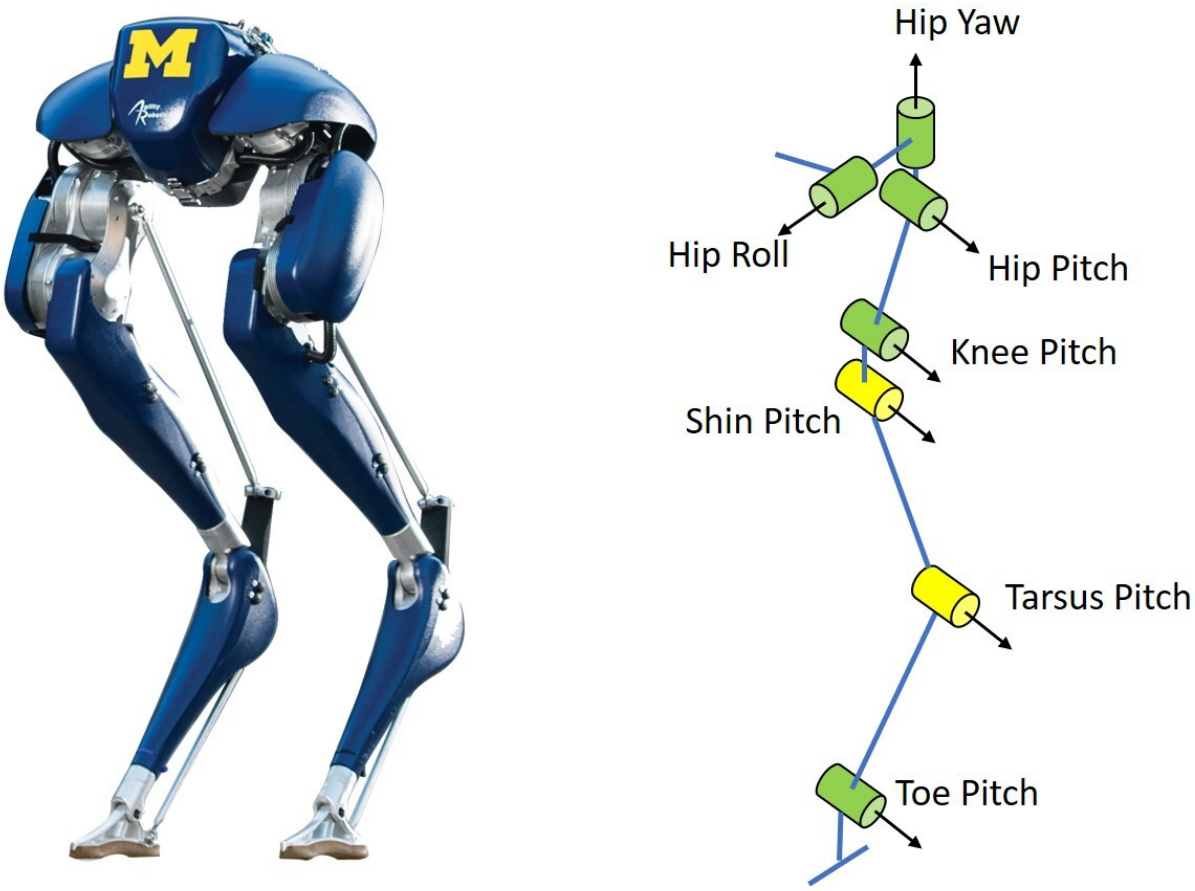


Figure 2.2: (Left) The Cassie Blue bipedal robot next to (Right) a schematic of the joints of the Cassie robot.

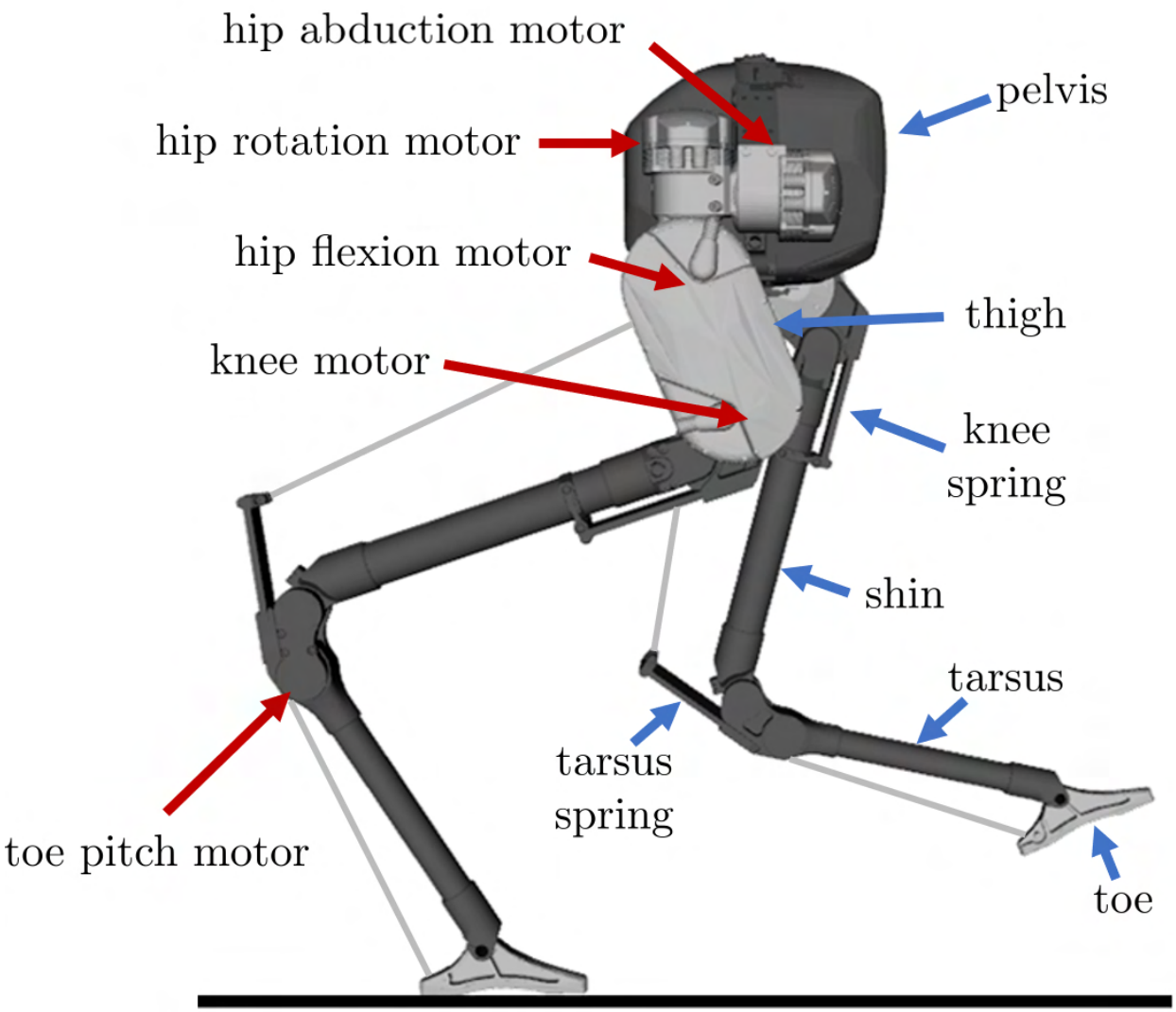


Figure 2.3: The joints and kinematic model of the Cassie robot.

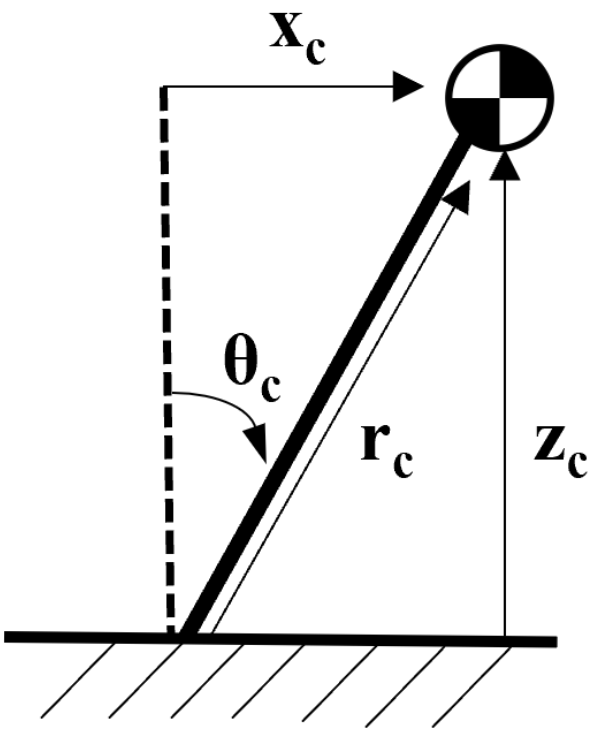


Figure 2.4: Schematic of an inverted pendulum to derive a variation on the ALIP model.

Chapter 3

Development of the Stair-Climbing Algorithm for Underactuated Bipedal Robots

This chapter unveils the creation of our novel stair-climbing algorithm for bipedal robots. Leveraging the Cassie biped's mathematical model and dynamic insights of the previous chapter, we craft a versatile algorithm to navigate the complexities of stair ascension. We detail the algorithm's evolution through virtual constraint-based control and MPC. These methods converge to establish a stable stair-climbing gait.

3.1 Background

The study of bipedal robot locomotion over stairs is not new. Several scholars, such as Fu et al. [63] and Caron et al. [64], have delved into this field by creating stair-walking controllers for fully actuated humanoid robots with 32 and 34 DoF respectively. In [65], the authors generated open-loop stair gaits for the 3D underactuated 20 DoF Cassie bipedal robot studied in this report; closed-loop control was not explored. In [36], the authors were able to apply human data of planned and unplanned downsteps on the Cassie biped in simulation. Our thesis seeks to further expand the capabilities of the Cassie biped by achieving an asymptotically stable periodic gait on stairs. Prior work by Siekmann et al. [38] made use of reinforcement learning to design a closed-loop controller for the Cassie bipedal robot, perceiving stair height as an unseen perturbation to the controller.

Although this achievement is noteworthy, the resulting gait appears to provoke severe impacts, potentially damaging the robot. In this thesis, we assume the robot is able to perceive terrain geometry at least one-step ahead, enabling the design of a controller that produces smoother locomotion. Dai et al. [39] approached the issue by developing a dynamic walking controller for constrained footholds (including on stairs) by regulating an underactuated robot's vertical CoM. We seek an alternative approach to stair climbing using the often-overlooked stance ankle motor as well as virtual constraint-based control. While the use of virtual constraints in legged locomotion is not new [66, 31, 35, 1, 67, 68, 69], the use of virtual constraints for a stair-climbing algorithm for a bipedal robot presents a novel use-case.

3.2 Novel Contributions

This thesis develops a controller that allows the Cassie biped to climb stairs. Novel contributions include the exploitation of a variation of the ALIP model that allows CoM height to vary within a step, and a novel combination of virtual constraint-based control and MPC to stabilize a stair-climbing gait.

If the ultimate goal is to have a bipedal robot navigate through cluttered environments, speed may not be the first priority. Rather, precision in balance is a necessity. We show the ability to modulate a robot's closed-loop behavior in real-time so as to smoothly handle stairs as well as reject perturbations on flat and inclined ground.

3.3 Control Design Rationale

Figure 3.1 summarizes how the novel stair-climbing controller of this thesis is developed. The Cassie biped has 20 DoF to control. This section breaks down how we chose to regulate these degrees of freedom.

During single support (one foot on the ground and the other free of contact), 9

DoF have holonomic constraints imposed on them: four from Cassie's springs (two springs on each leg), and five from the stance foot. Thus, we are left with 11 DoF to control and 10 actuators. The robot is therefore underactuated.

Previous work that has successfully achieved stable walking on level, inclined, and gently rolling terrain consistently used only nine of the ten actuators to achieve stable walking [31, 35], excluding the stance ankle motor. The stance ankle torque is not used in walking because the small ankle motor saturates easily on the real robot in the presence of disturbances, leading to falling. The remaining two uncontrolled degrees of freedom correspond to rotations of the robot about the stance foot in the sagittal and frontal planes and are stabilized via foot placement. As noted by Raibert in [70], if a robot's CoM spends more time in front of the stance foot than behind it, then it generally accelerates, and conversely, it decelerates. This property has been used by many authors to propose foot placement control algorithms [61, 71, 72, 73, 74, 75, 76] for stabilization of pendulum models.

We follow [31, 35] and use nine actuators to enforce nine virtual constraints, leaving two degrees of freedom uncontrolled. We adopt the foot placement strategy of [61] to stabilize the degree of freedom related to rotation about the stance foot in the frontal plane. Stairs offer limited geometry for sagittal foot placement and therefore foot placement in this plane is impractical. Instead, we use intelligent ankle torque control in a manner such that saturation will not destabilize the robot. This is developed in Section 3.6.

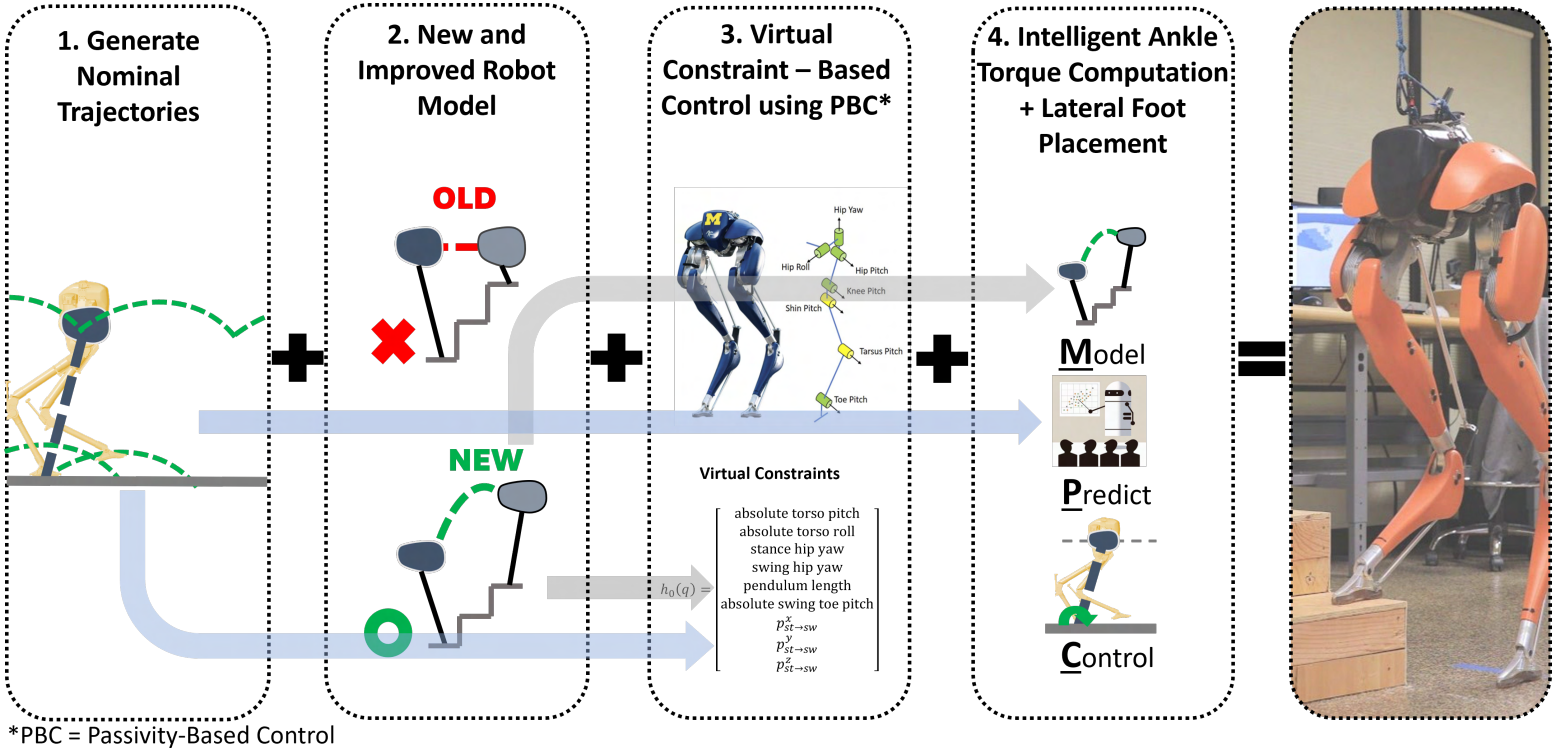


Figure 3.1: Diagram depicting the structure for the novel stair-climbing controller developed in this thesis.

3.4 Nominal Periodic Orbit

We utilized Fast Robot Optimization and Simulation Toolkit (FROST) [45] to generate nominal trajectories for Cassie, leveraging its advanced framework designed for optimizing and simulating the full-body dynamics of bipedal walking robots. FROST seamlessly integrates virtual constraints-based feedback controllers and employs a Wolfram Mathematica backend to symbolically generate expressions for multi-domain system dynamics and kinematics. These symbolic expressions are then translated into C/C++ code, compiled into *.MEX files under MATLAB, enhancing computational speed.

FROST conceptualizes dynamic bipedal walking as a hybrid system, combining both continuous phases and discrete transitions, represented by a Directed Graph. A notable feature is its use of state-of-the-art direct collocation approaches for gait optimization, ensuring swift and dependable convergence. The default control law of FROST relies on virtual constraints-based feedback controllers. This toolkit emerges as a versatile and efficient resource for researchers exploring bipedal robot control and dynamics optimization.

In our study, FROST was employed to generate multiple nominal trajectories for Cassie. Specifically, we crafted a nominal periodic orbit for Cassie ascending stairs, defining parameters such as step period ($T = 400$ ms), step height (20 cm, adhering to the regulated standard in the United States), and the "run of the step" (stair depth) at 25 cm (with no standardized value in the United States) [23, 77, 78, 79, 80]. Utilizing Bèzier curves [81], we approximated optimized trajectories for the nine virtual constraints (defined in Section 3.5), as well as the angular momentum and CoM angle crucial for MPC computation for ankle torque, as detailed in Section 3.6.

Furthermore, we generated trajectories for various scenarios, including marching in

place, walking forward, and transitioning from walking on flat ground to different inclines (4 degrees, 8 degrees, 15 degrees, and 20 degrees).

3.4.1 Generating Nominal Trajectories with FROST

Figure 3.2 illustrates the video outtakes capturing a nominal trajectory designed for stair climbing using FROST and a full-order model of the Cassie biped. The trajectory encompasses two steps, accommodating both right and left foot stance phases. The generation of this trajectory involved addressing several key constraints. Similar constraints were imposed when crafting the other nominal trajectories.

To prevent Cassie's feet from impacting the stairs, a constraint was introduced in which FROST is prohibited from allowing Cassie's feet to breach a specified radial distance (determined by stair geometry) around the desired foot placement. Additionally, a periodicity constraint was imposed at the conclusion of the second step, enabling the use of this trajectory to traverse an unlimited number of stairs through the simple repetition of the nominal trajectory.

Furthermore, a constraint was incorporated to minimize torque on the stance ankle motor. This strategic addition provides increased adaptability when applying the trajectory to an actual Cassie robot. Our hypothesis posited that a nominal trajectory minimizing stance ankle torque would translate to minimal ankle torque requirements for the physical robot when walking unperturbed, allowing for more room to overcome disturbances by increasing stance ankle torque output. This hypothesis was validated, as discussed in detail in Chapter 4 and Chapter 5.

In Figure 3.3, we present plots depicting the angular momentum and CoM angle, comparing the results computed by FROST for the nominal trajectories against the values obtained by the new variation of the ALIP model. As anticipated, the values are similar. This shows that the trajectories generated by FROST satisfy the ALIP model.

3.4.2 Bézier Curves

We translate the numerical values of the nominal trajectories generated by FROST and express them in the language of Bézier curves so that they can be more readily used and manipulated by the controller developed in this thesis. A Bézier curve is a mathematical representation of a curve defined by two or more "control points," which may be situated either on the curve itself or externally [2].

The order of a Bézier curve is determined by the number of control points it possesses. Specifically, the order equals the number of control points minus one. For instance, a linear curve (a straight line) involves two control points, a quadratic curve (a parabola) has three control points, and a cubic curve requires four control points. Irrespective of the number of control points, a Bézier curve is always contained within the "convex hull" of its control points—the minimal enclosed space that encompasses all control points.

A one-dimensional Bézier polynomial of degree M is expressed as follows:

$$b_i(s) := \sum_{k=0}^{M} \alpha_k^i \frac{M!}{k!(M-k)!} s^k (1-s)^{M-k} \tag{3.1}$$

where, $s \in [0,1]$, i denotes the i^{th} Bézier curve, M signifies the order of the curve, and α_k^i represents the k^{th} coefficient for the i^{th} Bézier curve, with $M+1$ coefficients for each curve.

When employing a Bézier curve, the function $x(q)$ of generalized coordinates may not necessarily span only the unit interval. To conform to the requirement that the independent variable s of the Bézier curve spans the unit interval $[0,1]$, normalization is often necessary:

$$s(q) := \frac{x(q) - x^+}{x^- - x^+}. \tag{3.2}$$

Here, x^- and x^+ denote the smallest and largest allowable values for x, respectively.

An interesting observation is that the partial derivative of the Bézier curve with respect to s is given by:

$$\frac{\partial b_i(s)}{\partial s} = \sum_{k=0}^{M-1} \left(\alpha_{k+1}^i - \alpha_k^i\right) \frac{M!}{k!(M-k-1)!} s^k (1-s)^{M-k-1}. \tag{3.3}$$

Figure 3.4 illustrates a Bézier curve of order 5, featuring six control points (the order plus one). The curve lies within the convex hull of these control points, which are equivalent to the Bézier curve coefficients α_k^i from Equation (3.1). Notably, the curve commences at $b(0) = \alpha_0$ and concludes at $b(1) = \alpha_5$. This alignment is intentional, as for all Bézier curves, $b_i(0) = \alpha_0^i$ and $b_i(1) = \alpha_M^i$, signifying that the i^{th} Bézier curve starts at the first coefficient α_0^i and concludes at the last coefficient α_M^i.

By incorporating Bézier curves into our controller for representing the nominal trajectories from FROST, we gain the ability to finely manipulate these trajectories through adjustments to their control points. For instance, to extend the time the swing foot takes to ascend and avoid contact with the upcoming stair, we strategically reposition the interior control points associated with the sagittal plane motion of the swing foot, moving them closer to the initial control point. This adjustment effectively prolongs the swing foot's retention of its sagittal position as it elevates in the $z-$ plane. The utilization of Bézier curves empowers us to make nuanced refinements to the nominal trajectories, such as this, without the need for the time-intensive task of regenerating trajectories through FROST.

3.5 Passivity-Based Control

Passivity-Based Control (PBC) is a powerful control strategy used to control nonlinear systems such as bipedal robots [82, 83, 84]. It has practical use for

hardware applications because it does not require an accurate model of the system. This is a key feature that adds a layer of robustness to shield from imperfect sensors and uncertain kinematic and dynamic properties within the robot.

We impose a spring constraint such that

$$J_s\ddot{q} + \dot{J}_s\dot{q} = -K_D^{\text{spring}} J_s\dot{q} - K_P^{\text{spring}} P_s^{\text{error}} \tag{3.4}$$

where P_s^{error} is the spring position error and K_D^{spring} and K_P^{spring} are user-defined derivative and proportional controller gains for the springs, respectively.

We additionally impose a non-slip constraint such that

$$J_{st}\ddot{q} + \dot{J}_{st}\dot{q} = 0. \tag{3.5}$$

From Equation (2.5), Equation (3.4), and Equation (3.5) we get

$$\tilde{D}f + \tilde{H} = \tilde{B}u_9 \tag{3.6}$$

where

$$\begin{aligned} \tilde{D} &= \begin{bmatrix} D & -J_{st}^\top & -J_s^\top \\ J_{st} & 0 & 0 \\ J_s & 0 & 0 \end{bmatrix}, \quad f = \begin{bmatrix} \ddot{q} \\ F_{st} \\ F_s \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_9 \\ 0 \\ 0 \end{bmatrix}, \text{ and} \\ \tilde{H} &= \begin{bmatrix} C\dot{q} + G - B_1 u_1 \\ \dot{J}_{st}\dot{q} \\ \dot{J}\dot{q} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -K_D^{\text{spring}} J_s\dot{q} - K_P^{\text{spring}} P_s^{\text{error}} \end{bmatrix}. \end{aligned} \tag{3.7}$$

We order the generalized coordinate vector q such that $q = [q_c \quad q_u]^\top$, where q_c are the controlled joints and q_u are the uncontrolled joints. We define $\lambda =$

$[q_u \quad F_{st} \quad F_s]^\top$ and partition Equation (3.7) such that

$$\begin{aligned} \tilde{D}_{11}\ddot{q}_c + \tilde{D}_{12}\lambda + \tilde{H}_1 &= \tilde{B}_1 u_9 \\ \tilde{D}_{21}\ddot{q}_c + \tilde{D}_{22}\lambda + \tilde{H}_2 &= \tilde{B}_2 u_9. \end{aligned} \tag{3.8}$$

That is,

$$\begin{bmatrix} \tilde{D}_{11} & \tilde{D}_{12} \\ \tilde{D}_{21} & \tilde{D}_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_c \\ \lambda \end{bmatrix} + \begin{bmatrix} \tilde{H}_1 \\ \tilde{H}_2 \end{bmatrix} = \begin{bmatrix} \tilde{B}_1 \\ \tilde{B}_2 \end{bmatrix} u_9. \tag{3.9}$$

We eliminate λ by using Schur Complement, resulting in

$$\bar{D}\ddot{q}_c + \bar{H} = \bar{B}u_9 \tag{3.10}$$

where

$$\begin{aligned} \bar{D} &= \tilde{D}_{11} - \tilde{D}_{12}\tilde{D}_{22}^{-1}D_{21} \\ \bar{H} &= \tilde{H}_1 - \tilde{D}_{12}\tilde{D}_{22}^{-1}\tilde{H}_2 \\ \bar{B} &= \tilde{B}_1 - \tilde{D}_{12}\tilde{D}_{22}^{-1}\tilde{B}_2. \end{aligned} \tag{3.11}$$

We define the output function as

$$y(x) = h_0(q) - h_d(q, {p_{sw}^x}^{des}, {p_{sw}^y}^{des}, {p_{sw}^z}^{des}, t) \tag{3.12}$$

where h_0 is the collection of virtual constraints and h_d provides the desired trajectories for the virtual constraints. In part due to precedent [1, 35, 61] and in part due to the new ALIP model defined in Section 2.3.2 that is being used for this

thesis, the virtual constraints are defined as follows:

$$h_0(q) = \begin{bmatrix} \text{absolute torso pitch} \\ \text{absolute torso roll} \\ \text{stance hip yaw} \\ \text{swing hip yaw} \\ \text{pendulum length} \\ p^x_{st\to sw} \\ p^y_{st\to sw} \\ p^z_{st\to sw} \\ \text{absolute swing toe pitch} \end{bmatrix} \tag{3.13}$$

where the pendulum length describes the vector r_c from the stance foot to the CoM and $p_{st\to sw}$ is the vector emanating from the stance foot and ending at the swing foot.

We design a passivity-based controller such that

$$\bar{D}\ddot{y} + (\bar{C} + K_D)\dot{y} + K_P y = 0 \tag{3.14}$$

where K_D and K_P are user-defined derivative and proportional controller gains, respectively. When designing the controller, we check that the decoupling matrix is full rank and we assume that the stance ankle torque is known. The required value of the ankle torque is developed in Section 3.6.

We can show that the error dynamics converge in Equation (3.14) using LaSalle's Invariance Principle, commonly known as LaSalle's Theorem.

LaSalle's Theorem *Consider a system described by the Ordinary Differential Equation (ODE) $\dot{x} = f(x)$, where x is the state vector and f is a vector field that defines the dynamics. Let $D \subset \mathbb{R}^n$ be a positively invariant set (i.e., if $x(t)$ is in D for some t, then $x(t)$ is in D for*

all $t \geq 0$) contained in a region Ω. Suppose there exists a continuously differentiable function $V : \Omega \to \mathbb{R}$ such that:

1. *$V(x)$ is positive definite (i.e., $V(x) > 0$ for all $x \neq 0$ in Ω),*
2. *$\dot{V}(x) \leq 0$ for all x in Ω (where $\dot{V}(x)$ is the time derivative of V along trajectories of the system),*
3. *$\dot{V}(x) = 0$ only on D.*

Then, every solution $x(t)$ of the system with initial conditions in Ω approaches the largest invariant set contained in D as $t \to \infty$. In other words, the trajectories of the system converge to the set D as time progresses [42].

We now show that the error dynamics converge in Equation (3.14) using LaSalle's Theorem. First, we construct a Lyapunov Function. From [82] and [31], we know the Lyapunov Function for an underactuated bipedal robot such as Cassie is

$$V = \frac{1}{2}\dot{y}^\top \bar{D}\dot{y} + \frac{1}{2}y^\top K_P y, \tag{3.15}$$

which is positive definite. That is, $V > 0$.

Taking the time derivative of Equation (3.15) yields

$$\dot{V} = \dot{y}^\top\big(-(\bar{C} + K_D)\dot{y} - K_P y\big) + \frac{1}{2}\dot{y}^\top \dot{\bar{D}}\dot{y} + \dot{y}^\top K_P y \tag{3.16}$$

$$\dot{V} = -\dot{y}^\top K_D \dot{y}. \tag{3.17}$$

From Equation (3.17), we know that $\dot{V}$ is negative semi-definite. That is, $\dot{V} \leq 0$. Using Equation (3.17), $\dot{V} \equiv 0$ implies $\dot{y} \equiv 0$, and hence $\ddot{y} \equiv 0$. Considering Equation (3.14), this further implies $y \equiv 0$. Thus, from LaSalle's Theorem, the error dynamics are stable.

3.6 Model Predictive Control using Quadratic Programming

Model Predictive Control (MPC) is a practical approach to controlling a robot through cluttered environments [85, 86, 87, 88]. By letting the robot "see ahead of time"—much like humans do when similarly moving through cluttered environments—it is easier to plan control actions that ensure the robot does not fall.

The idea of using MPC for bipedal locomotion on non-flat terrain is not new. In [89, 90], the authors generated trajectories for bipedal locomotion on stairs using MPC. Meanwhile, in [24], the authors implemented an MPC-based stair walking controller on a planar robot that had 5 DoF. This thesis seeks to use MPC to compute torque values for the often neglected stance ankle motor to help bipedal robots climb stairs.

The premise of *Model Predictive Control* is to use a *model* of a system to *predict* how the system will evolve over an interval of time to determine an optimal set of *control* inputs to achieve a desired goal state.

3.6.1 Discrete-time Model Formulation

We define the state of Equation (2.13) to be $x(t) = [\theta_c(t) \quad L(t)]^\top$ and convert the differential equation into a discrete-time model via

$$\dot{x}(t) \approx \frac{x(t + \Delta t) - x(t)}{\Delta t}. \tag{3.18}$$

We let $x_k = x(k\Delta t)$ so that the model can be expressed as

$$x_{k+1} = Ax_k + b_k u_k \tag{3.19}$$

where

$$A_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \Delta t \begin{bmatrix} 0 & \frac{1}{mr_c(k\Delta t)^2} \\ mgr_c(k\Delta t) & 0 \end{bmatrix}$$

$$b_k = \Delta t \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$u_k = \tau(k\Delta t).$$

While b_k does not vary with time, it is convenient to know which control signal it is distributing in the formulas below. Equation (3.19) defines our *model* for MPC.

3.6.2 Predictive Step

Given our model as well as values for our current state at time k, we can calculate the state $k+N$ at the end of a *horizon* of length N,

$$\begin{aligned} x_k &= \text{ given or measured from the robot} \\ x_{k+1} &= A_k x_k + b_k u_k \\ x_{k+2} &= A_{k+1} x_{k+1} + b_{k+1} u_{k+1} \\ &= A_{k+1} A_k x_k + A_{k+1} b_k u_k + b_{k+1} u_{k+1} \\ &\vdots \\ x_{k+N} &= A_{k+N-1} \cdots A_k x_k + A_{k+N-1} \cdots A_{k+1} b_k u_k + \\ &\qquad \cdots A_{k+N-1} b_{k+N-2} u_{k+N-2} + b_{k+N-1} u_{k+N-1}. \end{aligned} \tag{3.20}$$

For compactness, we rewrite this as

$$x_{k+N} = S_k x_k + \Gamma_k u_k^{seq} \tag{3.21}$$

where

$$\begin{aligned} S_k &:= A_{k+N-1} \cdots A_k \\ u_k^{\text{seq}} &:= \begin{bmatrix} u_k & u_{k+1} & \cdots & u_{k+N-2} & u_{k+N-1} \end{bmatrix}^\top \end{aligned} \tag{3.22}$$

and Γ_k can be computed recursively by

$$\begin{aligned} B_k &:= b_k \\ B_{k+j} &:= \begin{bmatrix} A_{k+j} B_{k+j-1} & b_{k+j} \end{bmatrix}, 1 \leq j \leq N-1 \\ \Gamma_k &:= B_{k+N-1}. \end{aligned} \tag{3.23}$$

We note that Γ_k is a $2 \times N$ matrix. For $N \geq 2$, it can be checked that Γ_k is full rank, that is, $\det(\Gamma_k \cdot \Gamma_k^\top) \neq 0$.

With this predictive model, we seek to compute u_k^{seq} such that

$$x_{k+N}^{\text{des}} = S_k x_k + \Gamma_k u_k^{\text{seq}} \tag{3.24}$$

where we'll select N to correspond to the duration of one robot step (i.e., a prediction horizon of 400 ms) and we'll choose x_{k+N}^{des} to be the corresponding value on the nominal periodic orbit at time $t = (k+N)\Delta T, \mod T$, where $T = 400$ms is the step period.

3.6.3 Control Computation

To minimize the torque sequence u_k^{seq} such that the dynamics hold, we implement a Quadratic Program (QP) [91]. A QP is a type of mathematical optimization problem that involves optimizing a quadratic objective function subject to linear equality and inequality constraints. The general form of a QP is as follows:

$$\text{Minimize: } \frac{1}{2} x^T P x + q^T x$$

$$\text{Subject to: } Gx \leq h$$

$$Ax = b$$

where,

x is the vector of variables to be optimized.

P is a symmetric positive-semidefinite matrix.

q is a vector.

G is a matrix defining linear inequality constraints.

h is a vector specifying the upper bounds on the inequality constraints.

A is a matrix defining linear equality constraints.

b is a vector specifying the right-hand side of the equality constraints.

The objective function in a QP is quadratic, and the constraints are typically linear. The solution to a QP is a vector x that minimizes the objective function while satisfying the given constraints.

In our application, we implement a QP and arrive at the following optimization problem:

$$\begin{aligned} &\min_{u_{seq}} \left[u_{seq}^T \;\; H(t)u_{seq} \quad + (x - x^{\text{des}})^\top Q(t)(x - x^{des}) \right] \\ &\textbf{subject to} \\ &\Gamma_k u_k^{\text{seq}} = x_{k+N}^{\text{des}} - S_k x_k \\ &u_{min} < u_k < u_{max} \end{aligned} \tag{3.25}$$

where $H(t)$ and $Q(t)$ are weighting matrices, and u_{min} and u_{max} are the lower and upper bounds imposed on the torque input, respectively. We select $H(t)$ and $Q(t)$

such that values toward the end of the step are weighted more, with the value at impacts being weighted the most heavily.

Remark: Using a QP is a common approach to solving the optimization problem posed from MPC. We discuss a naïve approach in Appendix C that we first implemented in the ideal simulator (discussed in Appendix E) as a proof-of-concept for this general control strategy.

3.7 Lateral Stabilization of the Robot

We stabilized the lateral motion of the Cassie biped by using the angular momentum-based foot placement strategy developed in [61], but with the new ALIP model derived in Section 2.3. This section derives the foot placement strategy for the new ALIP model. Unlike the previous version of the ALIP model, the new ALIP model does not have a closed-form solution. Thus, a numerical approach must be implemented for a foot placement strategy.

The strategy to compute lateral foot placement numerically is outlined as follows:

1. Use Euler method for numerical integration to estimate the angular momentum at the end of the **next** step, **before** impact, for two lateral foot positions $y^1_{\text{st}\to\text{sw}}$ and $y^2_{\text{st}\to\text{sw}}$, where $y^1_{\text{st}\to\text{sw}}$ and $y^2_{\text{st}\to\text{sw}}$ are close in value. We will call these two angular momentum values L_1 and L_2.

2. Define a desired angular momentum L_{des} that we assume to be approximately along the line between Point 1 $(y^1_{\text{st}\to\text{sw}},\ L_1)$ and Point 2 $(y^2_{\text{st}\to\text{sw}},\ L_2)$.

3. Using 2-Dimensional (2D) or 3-Dimensional (3D) linear interpolation, find y_{des}, the lateral foot placement position corresponding to L_{des}.

3.7.1 Euler Integration

Euler integration, also known as the Euler method, is a numerical technique for approximating the solution of an Ordinary Differential Equation (ODE) with an initial value problem [92]. The Euler method proceeds by discretizing the time domain into small steps and approximating the solution at each step. The iterative formula for Euler integration is:

$$x_{n+1} = x_n + \Delta t \cdot f(t_n, x_n) \tag{3.26}$$

where x_{n+1} is the approximation of the solution at the next time step, x_n is the solution at the current time step, Δt is the step size representing the time interval between consecutive steps, $f(t_n, x_n)$ is a given first-order ODE function evaluated at time n, and t_n is the current time.

The method starts with an initial value x_0 and t_0, and subsequent values are computed using the iterative formula. The smaller the step size Δt, the more accurate the approximation, but it also increases the computational cost.

In our application, we determine our initial values x_0 and t_0 as being the current state of the system $x_0 = [\theta_c(t_0) \quad L(t_0)]^\top$ at time t_0. We use Equation (3.26) to compute the state of the system at every time step Δt until the end of the **first** step of a period length $t = T$, **before** impact, where $f(t_n, x_n)$ is defined as the system dynamics from the new ALIP model from Equation (2.13). We use an impact map (derived in Section 3.7.2) to compute the system dynamics **after** impact of the **first** step. These state values are equivalent to the state values at the beginning of the **next** step. Using these post-impact values as initial values, we then implement Euler integration again to compute the state of the system at every time step from the beginning of the **next** step until the end of the step **before** impact.

3.7.2 Impact Map for the New ALIP Model

To compute the state of the system **after** impact of the **first** step, we develop an impact map for the new ALIP model. This new impact model for the reduced order model is separate from the impact map for the full-order Cassie model developed in Section 2.4.

Figure 3.5 illustrates the relevant variables to derive the new impact map. Let Point A be the position where the swing foot touches the ground and Point B be the point where the stance foot touches the ground. Using the angular momentum about Point B before impact L_B^-, we can compute the angular momentum about Point A after impact L_A^+. From the conservation of angular momentum, we know that the angular momentum about Point A before impact L_A^- is equivalent to the angular momentum about Point A after impact L_A^+. That is,

$$L_A^- = L_A^+. \tag{3.27}$$

Using the *angular momentum transfer formula* [31], we compute L_B^- from L_A^-,

$$L_A^- = L_B^- + P_{\text{st} \to \text{sw}} \times m v_c \tag{3.28}$$

where $P_{\text{st} \to \text{sw}} = [P^x_{\text{st} \to \text{sw}} \quad P^z_{\text{st} \to \text{sw}}]^\top$ is the vector from the stance to swing foot, m is the mass of the robot, and v_c is the CoM velocity. In the x-z plane, the position of the CoM is defined as

$$p_c = r_c^- \begin{bmatrix} \sin(\theta_c^-) \\ \cos(\theta_c^-) \end{bmatrix}. \tag{3.29}$$

Taking the derivative Equation (3.29) yields the velocity of the CoM,

$$\frac{d}{dt}p_c = v_c = \begin{bmatrix} r_c^- \cos(\theta_c^-) \cdot \dot{\theta}_c^- + \dot{r}_c^- \sin(\theta_c^-) \\ -r_c^- \sin(\theta_c^-) \cdot \dot{\theta}_c^- + \dot{r}_c^- \cos(\theta_c^-) \end{bmatrix}. \tag{3.30}$$

From Equation (3.27), Equation (3.28), and Equation (3.30), we compute

$$\begin{aligned} L_A^- = L_A^+ = L_B^- + m \Bigg[& P^z_{\text{st} \to \text{sw}} \Big(r_c^- \cos(\theta_c^-)\dot{\theta}_c^- + \dot{r}_c^- \sin(\theta_c^-) \Big) - \\ & P^x_{\text{st} \to \text{sw}} \Big(- r_c^- \sin(\theta_c^-)\dot{\theta}_c^- + \dot{r}_c^- \cos(\theta_c^-) \Big) \Bigg]. \end{aligned} \tag{3.31}$$

Using trigonmetry, we determine the value of the CoM angle after impact θ_c^+ to be

$$\theta_c^+ = \arccos\left(\frac{r_c^- \cos\theta_c^- - P^z_{\text{st} \to \text{sw}}}{r_c^+} \right). \tag{3.32}$$

Equation (3.31) and Equation (3.32) form our impact map. That is, the state of the system after impact is defined as

$$x^+ = \begin{bmatrix} \theta_c^+ \\ L^+ \end{bmatrix} = \begin{bmatrix} \arccos\left(\frac{r_c^- \cos\theta_c^- - P^z_{\text{st} \to \text{sw}}}{r_c^+} \right) \\ L_B^- + m \Bigg[P^z_{\text{st} \to \text{sw}} \Big(r_c^- \cos(\theta_c^-)\dot{\theta}_c^- + \dot{r}_c^- \sin(\theta_c^-) \Big) - \\ P^x_{\text{st} \to \text{sw}} \Big(- r_c^- \sin(\theta_c^-)\dot{\theta}_c^- + \dot{r}_c^- \cos(\theta_c^-) \Big) \Bigg] \end{bmatrix}. \tag{3.33}$$

3.7.3 Linear Interpolation

Figure 3.6 depicts the 2D linear interpolation method to compute the lateral foot placement. From this, we see that we can compute the desired lateral foot placement y_{des} by

$$y_{\text{des}} = y^1_{\text{st} \to \text{sw}} + \frac{L_{\text{des}} - L_1}{\text{slope}}, \tag{3.34}$$

where

$$\text{slope} = \frac{L_2 - L_1}{y^2_{\text{st}\to\text{sw}} - y^1_{\text{st}\to\text{sw}}}. \tag{3.35}$$

Similarly, one could use CoM angle, θ, instead of angular momentum to compute lateral foot placement. That is,

$$y_{\text{des}} = y^1_{\text{st}\to\text{sw}} + \frac{\theta_{\text{des}} - \theta_1}{\text{slope}}, \tag{3.36}$$

where

$$\text{slope} = \frac{\theta_2 - \theta_1}{y^2_{\text{st}\to\text{sw}} - y^1_{\text{st}\to\text{sw}}}. \tag{3.37}$$

Using just angular momentum or CoM angle alone to compute lateral foot placement is 2D interpolation. Linear interpolation in three dimensions utilizes both angular momentum and CoM angle together to compute lateral foot placement. The 3D linear interpolation computation for lateral foot placement is computed as follows:

$$\begin{bmatrix} y_{\text{des}} \\ \theta_{\text{des}} \\ L_{\text{des}} \end{bmatrix} = \begin{bmatrix} y_1 \\ \theta_1 \\ L_1 \end{bmatrix} + \frac{\sqrt{(y_{\text{des}} - y^1_{\text{st}\to\text{sw}})^2 + (\theta_{\text{des}} - \theta_1)^2 + (L_{\text{des}} - L_1)^2}}{||v||} \cdot v, \tag{3.38}$$

where

$$v = \begin{bmatrix} y^2_{\text{st}\to\text{sw}} - y^1_{\text{st}\to\text{sw}} \\ \theta_2 - \theta_1 \\ L_2 - L_1 \end{bmatrix}. \tag{3.39}$$

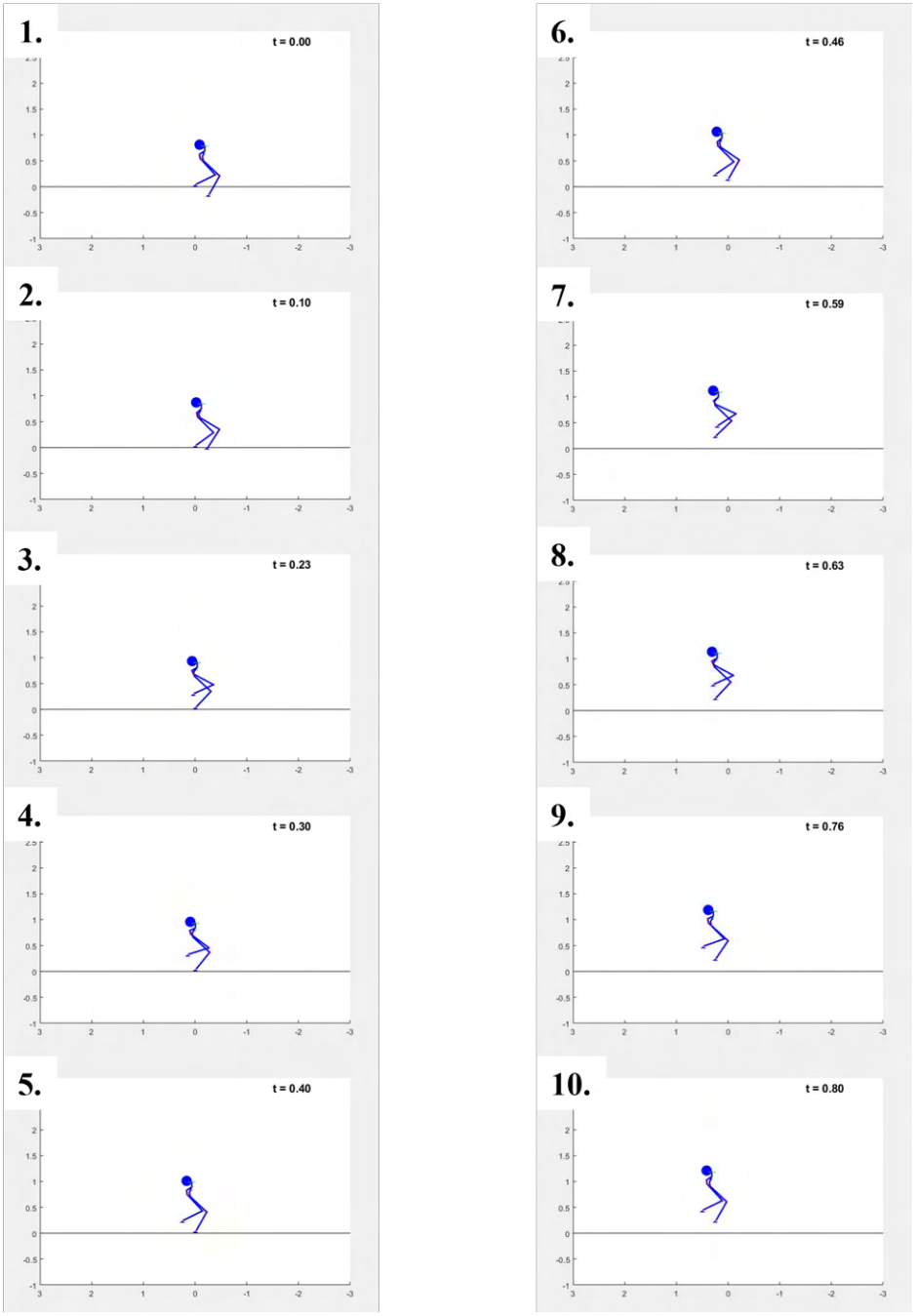


Figure 3.2: Outtakes of the stair-climbing nominal trajectory generated by FROST using a full-order model of the Cassie biped.

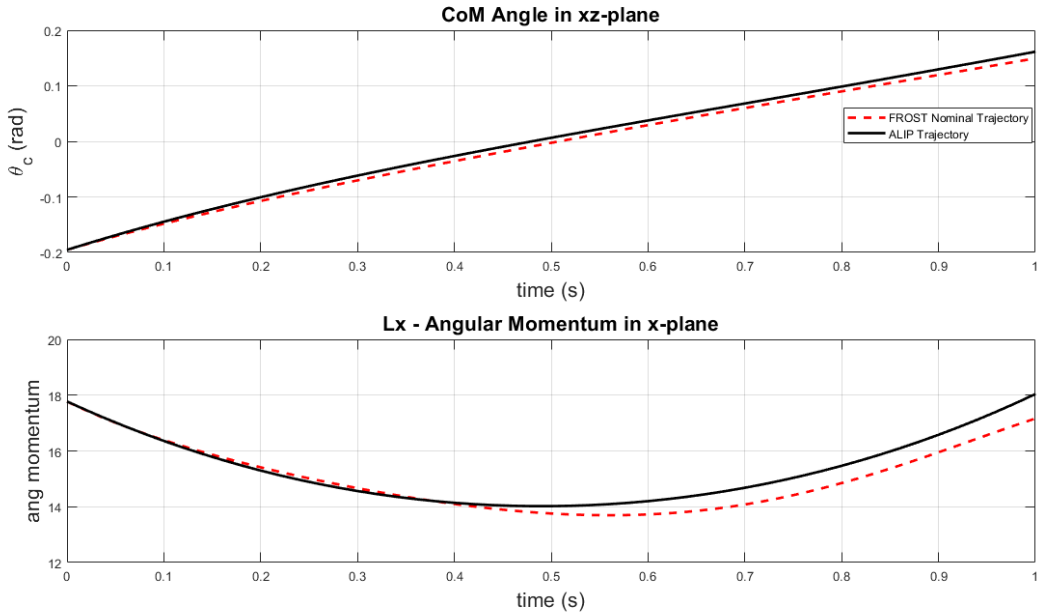


Figure 3.3: Comparison of the nominal trajectory generated by FROST and ALIP model of angular momentum and CoM angle.

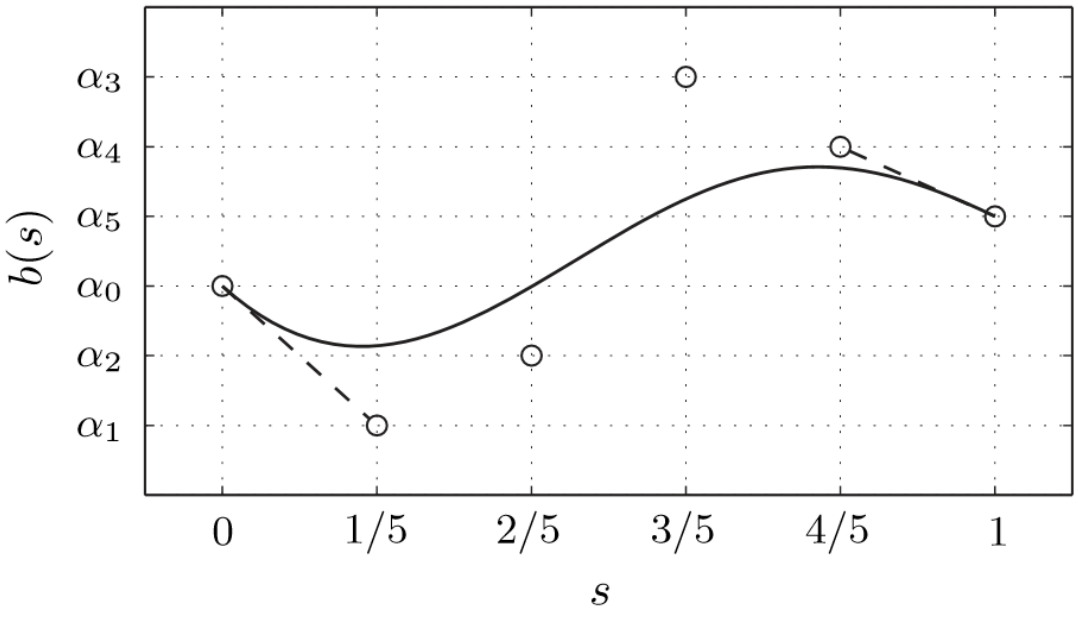


Figure 3.4: Graph of a Bézier Curve of Order 5 [2].

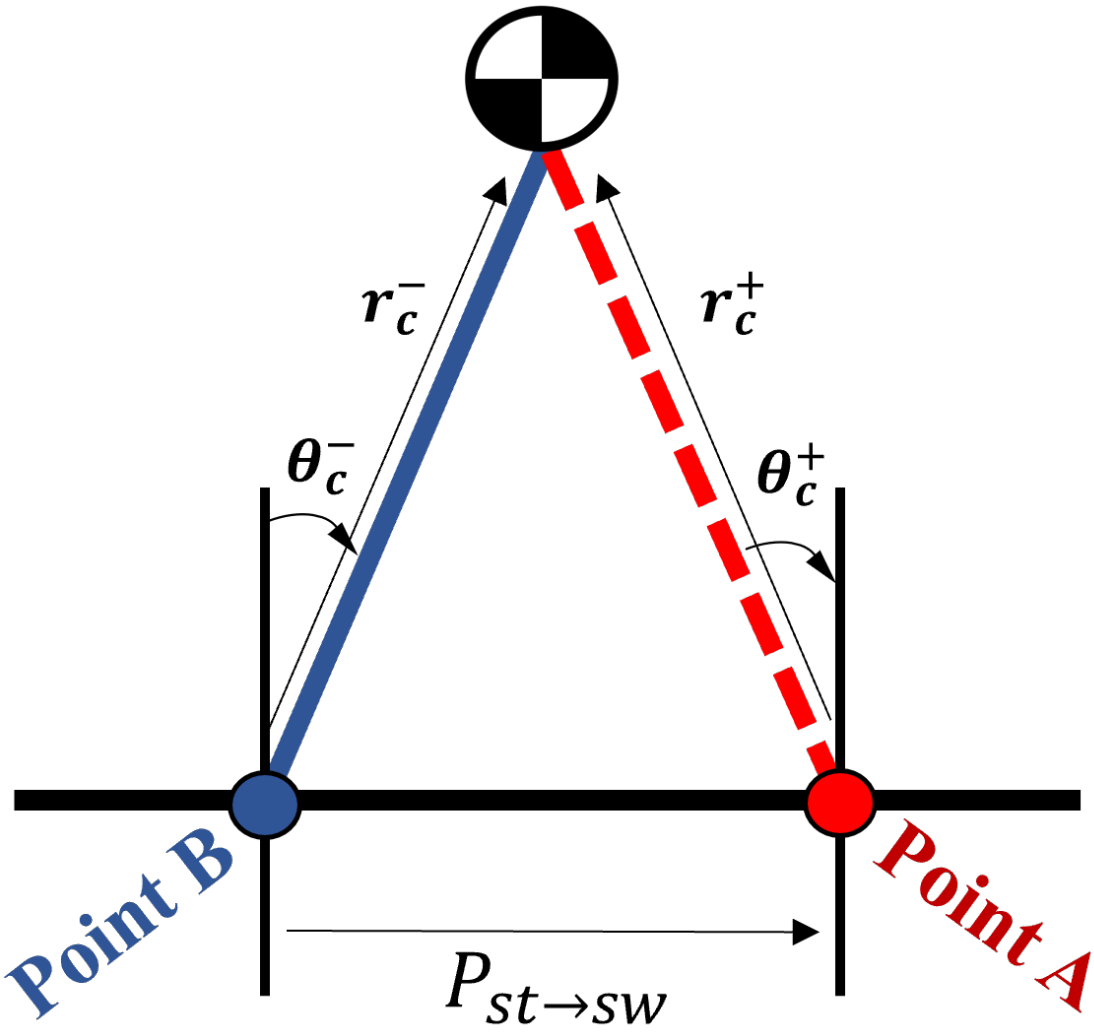


Figure 3.5: Schematic of the inverted pendulum to derive the impact map for the new variation on the ALIP model.

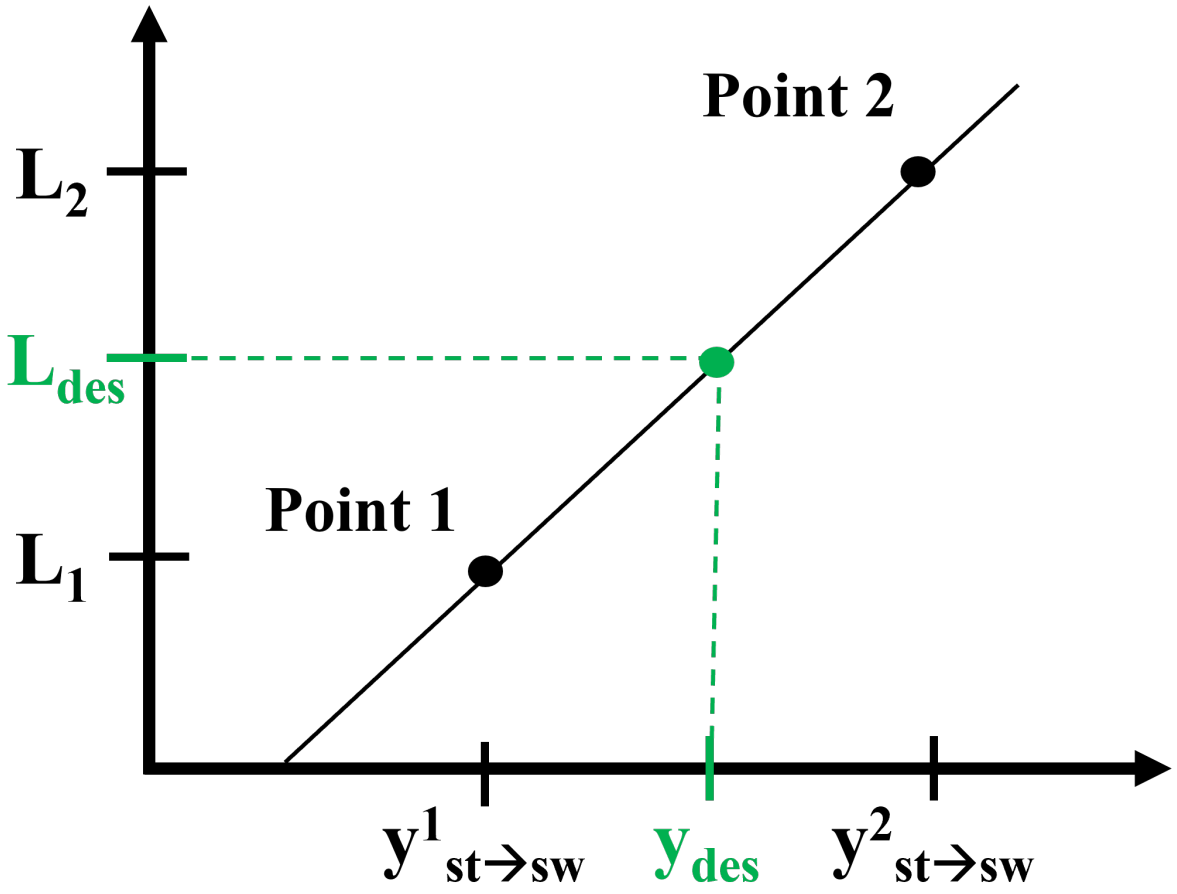


Figure 3.6: Graph depicting linear interpolation to compute lateral foot placement using angular momentum.

Chapter 4

Insights from SimMechanics - Simulation Results and Discussion

This chapter discusses the implementation of the controllers from Sections 3.5, 3.6 and 3.7 on the 20 DoF simulation model of the Cassie robot using Matlab and Simulink in the SimMechanics simulation environment. Unlike in the ideal simulator (discussed in Appendix E), the SimMechanics incorporates Cassie's stiff springs in the full-order model. Moreover, we implement state estimation using a Kalman filter to fuse accelerometer and encoder data [93, 94, 95], exactly as we would on hardware, for a more faithful representation of the physical Cassie robot.

Figure 4.1 and Figure 4.2 show the Cassie robot in the SimMechanics environment on stairs. Note the direction of the positive $x-$ and $z-$axes, which means that a *negative rotation* about the $y-$axis corresponds to *walking up* the stairs. This is an important observation for interpreting later plots.

4.1 Walking on Flat Ground

As a first check, we evaluated our controller on flat ground. We know from previous work [70] that foot placement alone on flat ground is enough to stabilize the system. Removing foot placement in the sagittal plane and instead using a fixed step length value (that is, setting the desired swing foot position to a predefined nominal value, similar to what needs to happen on stairs where the sagittal step length is constrained to a constant) results in an unstable closed-loop system. We posited that using ankle torque would then stabilize the system.

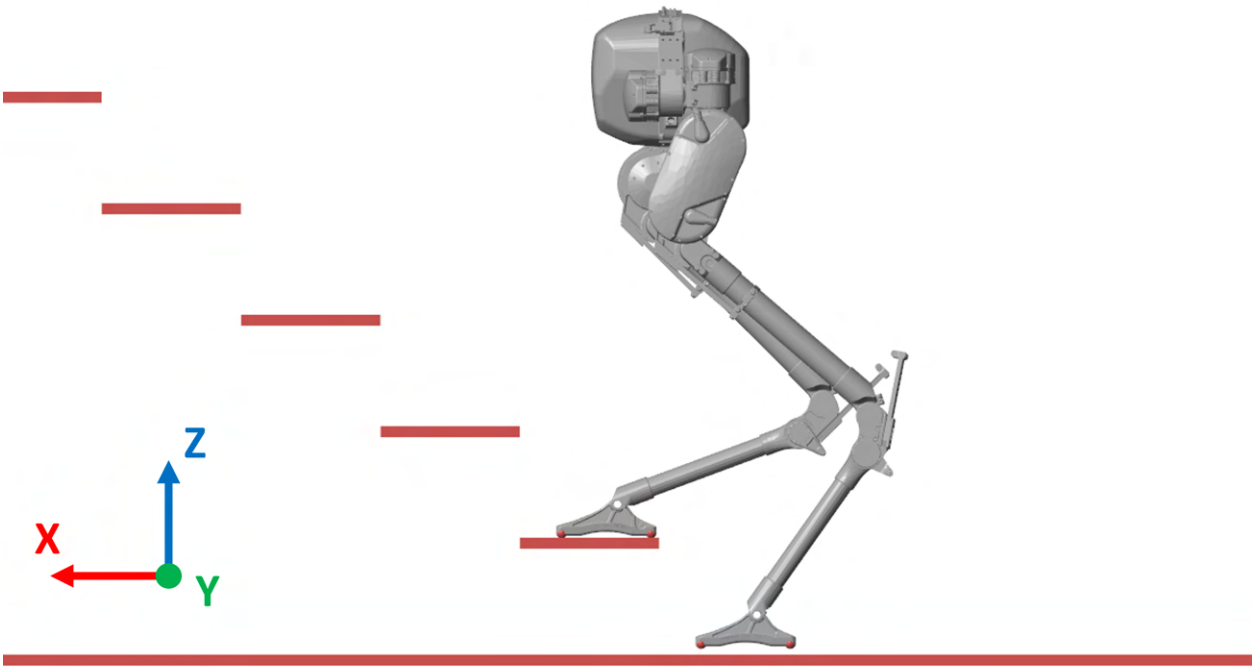


Figure 4.1: 3D model of the Cassie robot in the SimMechanics simulation environment.

Simulations showed this hypothesis to be correct. Turning off ankle torque while the robot walked with fixed step lengths resulted in the robot falling. Adding ankle torque control not only allowed the robot to walk continuously with fixed steps, but also made the system robust against perturbations.

Figure 4.3 depicts the outtakes for a simulation where the robot stands for the first two seconds, transitions to stepping in place for the next four simulation seconds, and then walks forward for the remainder of the simulation, activating the fixed step gait at the 12-second mark in the simulation time. Figure 4.4 shows two sets of plots corresponding to the total angular momentum and CoM angle for this simulation experiment. The first set of plots correspond to the simulation where ankle torque was not used during the fixed step portion of runtime. The second set of plots correspond to the simulation where ankle torque was used during fixed step. Note that the robot falls after just two steps when ankle torque is not engaged during the fixed step gait. This is because the fixed step trajectory does not allow the robot to maintain a periodic angular momentum trajectory, causing the angular momentum to lag behind the desired nominal trajectory and eventually falling. Absent of the intelligent foot placement method that could ensure that a angular momentum trajectory is followed, the system requires a force to maintain stability. Ankle torque supplies this necessary force to the system, pushing the robot back on to the nominal trajectory.

In Figure 4.5 and Figure 4.6, we demonstrate the robustness of our ankle torque controller. Following the same gait transitions as aforementioned, we perturb the system at simulation time $t = 3$ seconds (while the robot is walking in place) and $t = 14$ seconds (while the robot is walking forward in fixed steps) by reducing all motor torque inputs by one-fifth (1/5) of their desired value for 50 milliseconds. The perturbations resulted in a disturbance equivalent to a shift of 0.1 rad in the CoM angle and 5 kg-m^2/sec in angular momentum. In both cases, ankle torque

control was able to prevent a fall and return the robot to a periodic gait. In the absence of ankle torque, the robot falls.

4.2 Walking up Stairs

At each step, the swing foot is regulated to place the new stance foot near the center of the stair's tread; without this, small errors accumulate and result in the robot not respecting the stair's geometry. In simulations, this is straightforward to achieve. In future experiments, we'll use the perception system design for Cassie in [96, 97, 98, 99].

Using the passivity based controller of Section 3.5 alone to enforce fixed step lengths, without other control in the sagittal plane, resulted in the robot taking two steps and then falling backward. Activating the MPC controller for ankle torque resulted in the 20 DoF simulation model being able to walk an unbounded number of steps as depicted in [100].

Figure 4.7 shows the stance ankle torque inputs calculated via the MPC approach throughout the simulation period. We enforced a stance ankle torque limit of ± 23 Nm in the quadratic program solver. This value was decided based on the max torque limit of the ankle motor and the gear ratio of 50. Throughout the simulation, the stance ankle torque is predominantly negative, which means it is "pushing" in the direction of motion. Without the additional ankle torque, the robot falls backward, which results in a positive rotation about the $y-$axis.

Figure 4.8 shows the angular momentum and CoM angle as the robot walks up 10 stairs. The plots show both the nominal trajectory that was used to set the desired values for the MPC when determining stance ankle torque, as well as the actual simulated values. Note that even though the simulated trajectory is not exactly following the nominal trajectory, it is still able to achieve a stable periodic orbit. The optimized nominal trajectory was developed on a model of the Cassie

biped that does not factor in Cassie's springs. We applied our controller on a full order model of the Cassie biped in the SimMechanics simulation environment that includes Cassie's springs as a more faithful representation of the hardware model. Furthermore, we approximate Cassie's states using a Kalman Filter, exactly as we would on hardware, which adds more noise to the system. In the presence of all of these uncertainties and perturbations, our controller is still able to achieve a stable walking gait up stairs. This is discussed further in the next section.

4.3 Discussion

The nominal trajectory used for stair climbing was designed with FROST [45] using a model of Cassie that does not factor in the springs. In effect, the springs, therefore, act as perturbations to the system that the MPC-generated ankle torque must accommodate and overcome at each impact.

At impact, the relatively stiff springs in the stance leg come into play, leading to oscillations in the "knee joint" that are not present in the controller design model. This leads to the short-duration spikes in ankle torque seen in Figure 4.7. To confirm this is the source of the torque spikes, we show in Figure 4.9 a simulation of the planar nonlinear ALIP model in Equation (2.13) in closed-loop with the identical controller used on the full-order model of Cassie over a horizon length $N = 5T$. As expected, we achieved near perfect tracking with this simplified model compared to the poorer tracking on the full order model shown in Figure 4.8. Figure 4.10 shows the corresponding ankle torques for the simulation on the planar nonlinear ALIP model. Note the marginal torque values that evolve to become almost negligible by the fifth step in the horizon. This matches what we would expect. The optimized trajectories generated by FROST was computed by placing a constraint to minimize stance ankle torque. The planar nonlinear ALIP model is thus able to follow the optimized trajectory using minimal torque input.

While our controller has proven to be robust enough to handle the perturbations caused by the springs, we anticipate that enhanced robustness and agility will require a nominal trajectory that accounts for spring deflection. We can further improve the robustness of our controller by 1) using trajectories that are optimized over a model that factors in Cassie's springs, and 2) upgrading our ALIP model used in MPC to also factor in springs–in effect, using an Angular Momentum Spring-Loaded Inverted Pendulum (A-SLIP) model. With these changes, our novel control paradigm would not only be able to better handle perturbations to the system during flat ground walking and stair climbing, but also be able to used as the basis of a controller that can help a robot maintain balance while navigating through semi-cluttered environments.

4.4 Conculsions

We have presented a model-based control strategy for walking up a flight of stairs. The control strategy uses virtual constraints to control the robot's posture. A foot placement strategy ensures lateral stability because standard stair width does not impose any geometric limitations in the lateral direction. In the sagittal plane, however, stair tread depth makes foot placement impractical, and thus we adopted a strategy relying on ankle torque computed via a linearized time-varying model and MPC. Steady-state walking for a 20 DoF simulation model of the Cassie robot was demonstrated in SimMechanics for both flat ground walking and stair climbing.

The next step will be to apply this strategy on the physical Cassie robot, incorporating a perception system [96, 97, 98, 99], so that Cassie is able to navigate stairs autonomously.

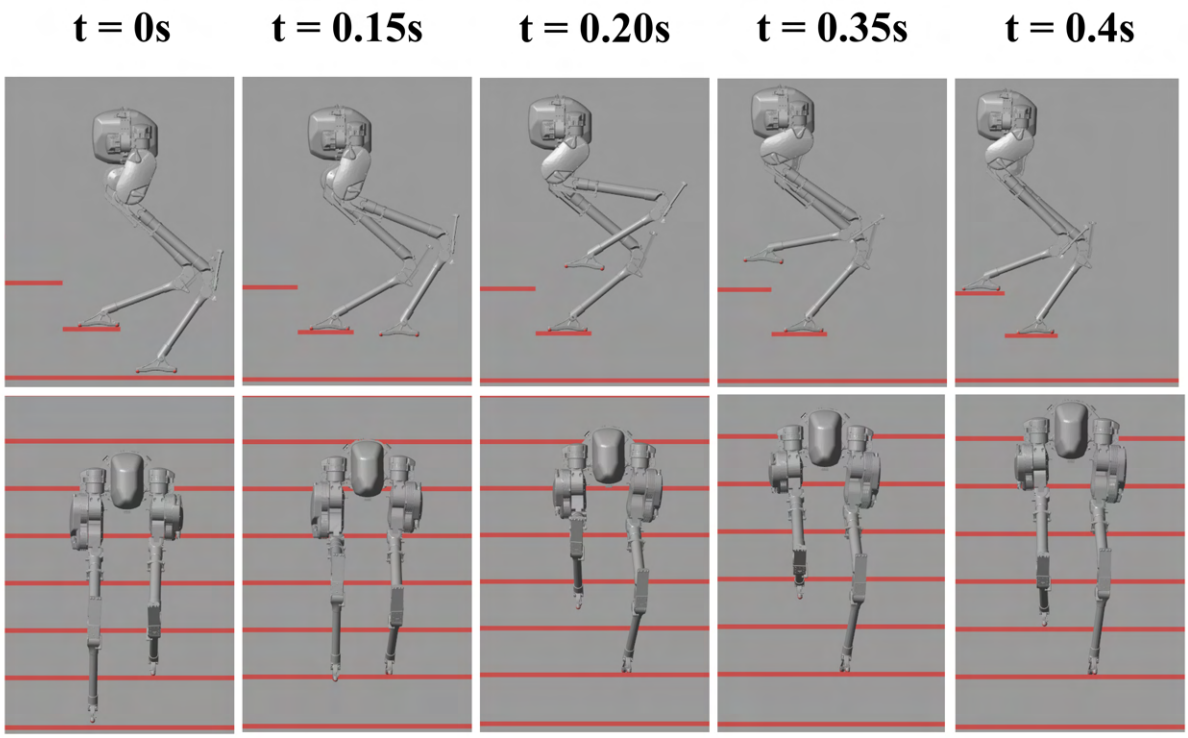


Figure 4.2: The underactuated Cassie biped walking up stairs in the SimMechanics Simulation environment.

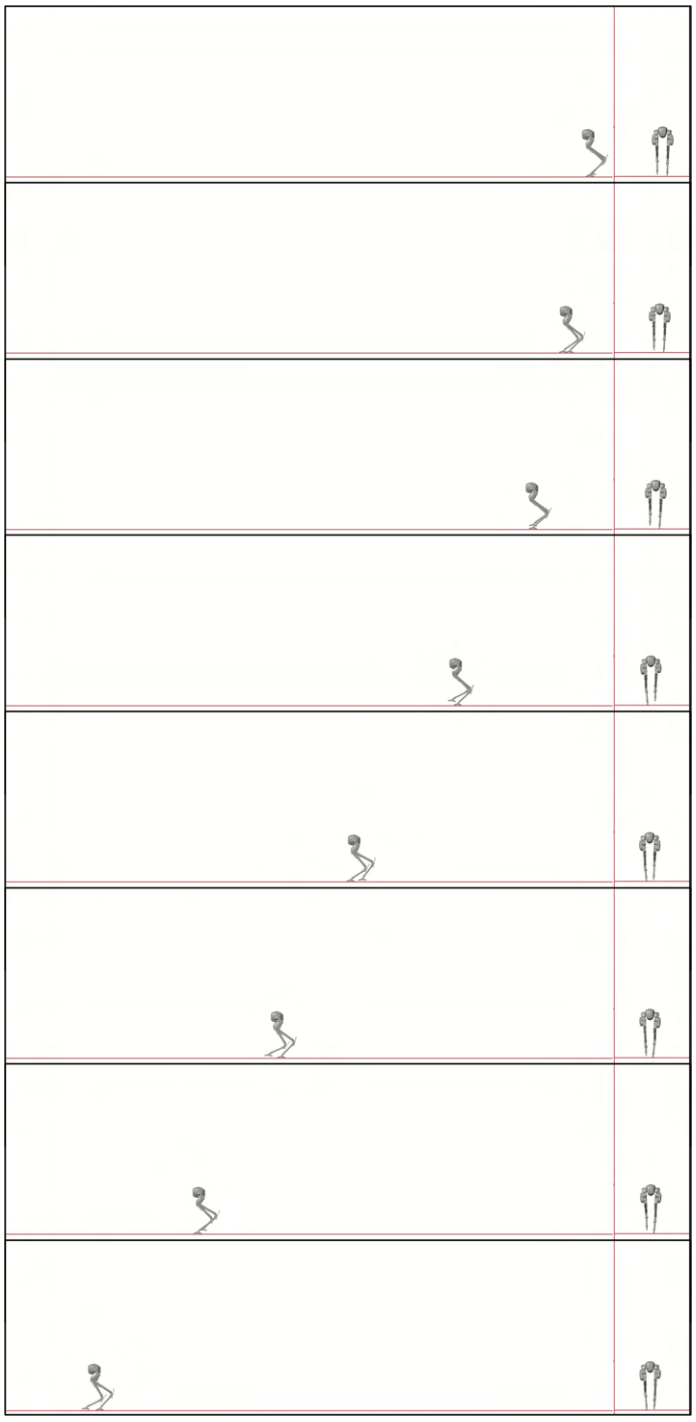


Figure 4.3: Outtakes of the underactuated Cassie biped for a SimMechanics simulation experiment. Cassie stands for the first two seconds, transitions to stepping in place for the next four simulation seconds, and then walks forward for the remainder of the simulation, activating the fixed step gait at the 12-second mark in the simulation time.

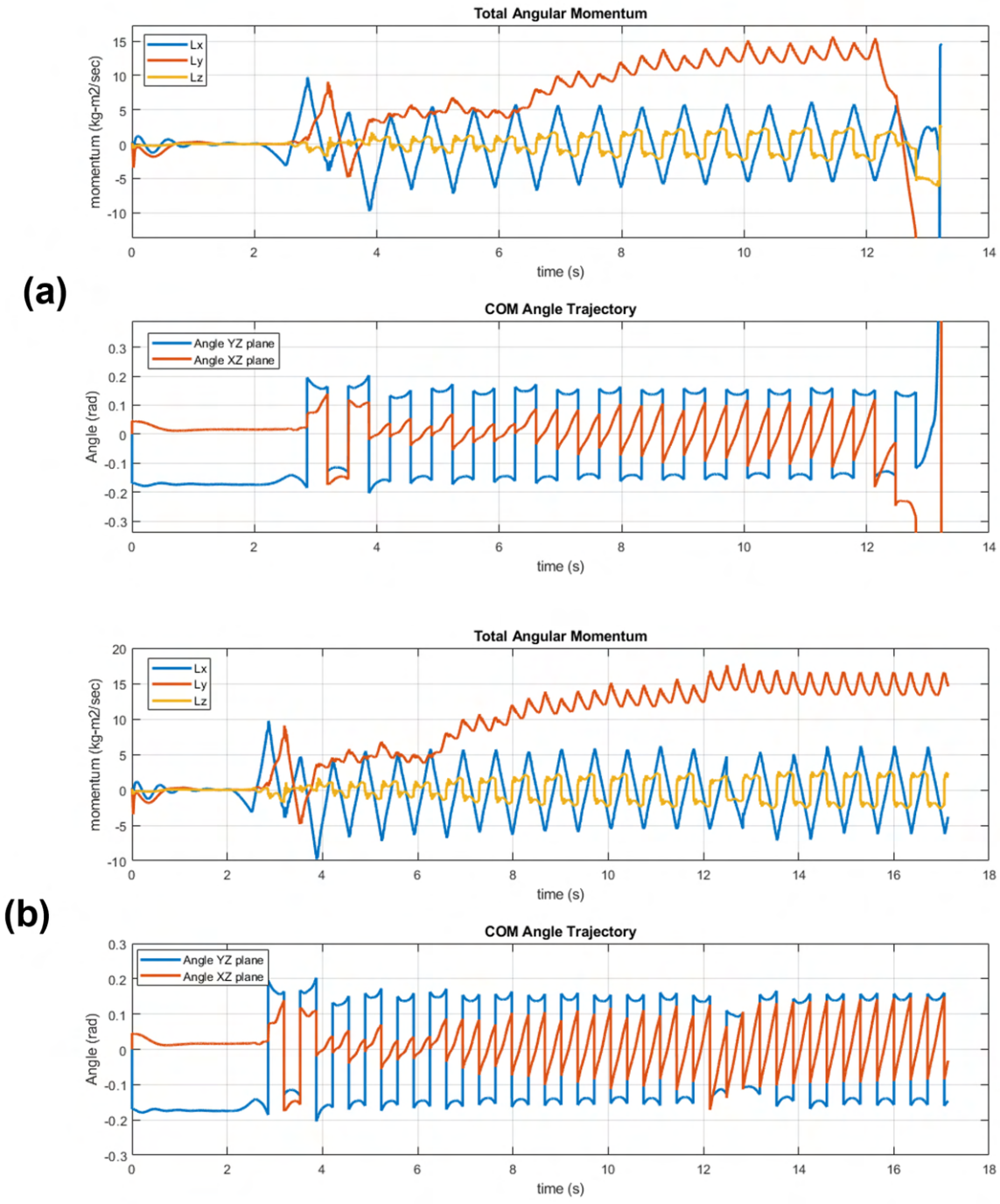


Figure 4.4: Angular momentum and CoM angle during simulation where robot stands for two seconds, steps in place for the next four seconds, and then is commanded to walk forward at 0.5 m/s for the remainder of the simulation runtime. Fixed step gait is turned on at the 12 second mark. Two test results are shown, (a) not using ankle torque during fixed step, and (b) using ankle torque during fixed step.

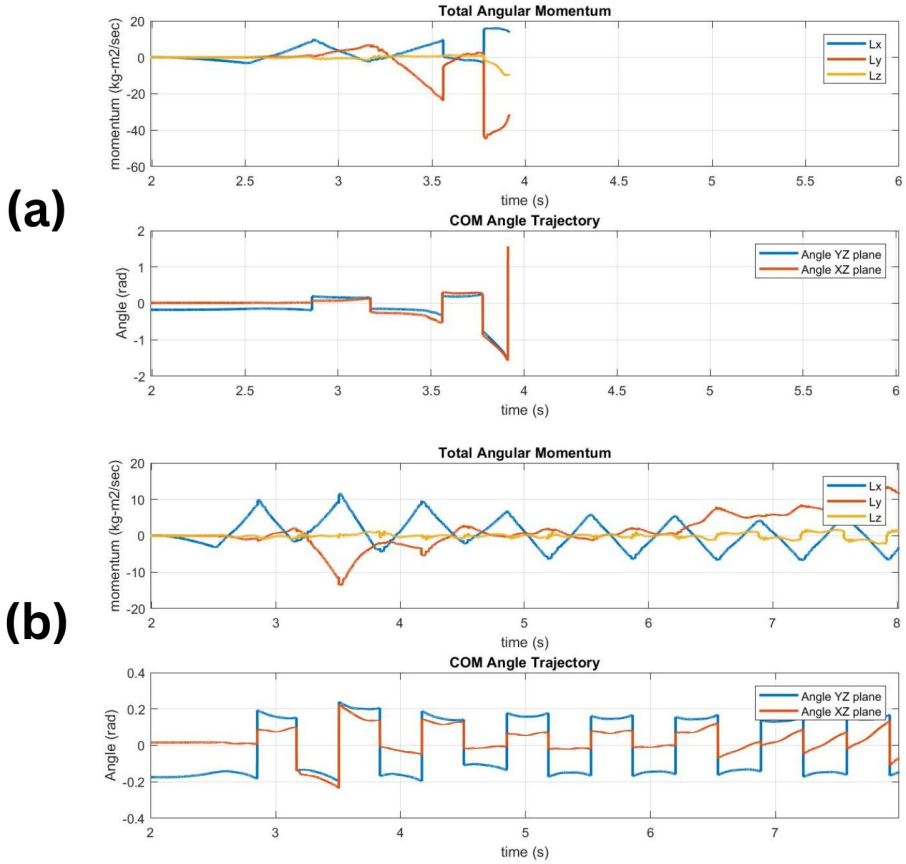


Figure 4.5: Angular momentum and CoM angle over time with and without ankle torque to stabilize marching in place with perturbations at $t = 3$ sec (a) without ankle torque and (b) with ankle torque. Note, that only the relevant time portion of the plot is shown $(2 < t < 8)$ to highlight the effects of the perturbation. In (a), there is no data after ~ 3.8 sec because the simulation fails at this time. Data continues until the end of the simulation for (b) because the robot is able to fully recover after the perturbation.

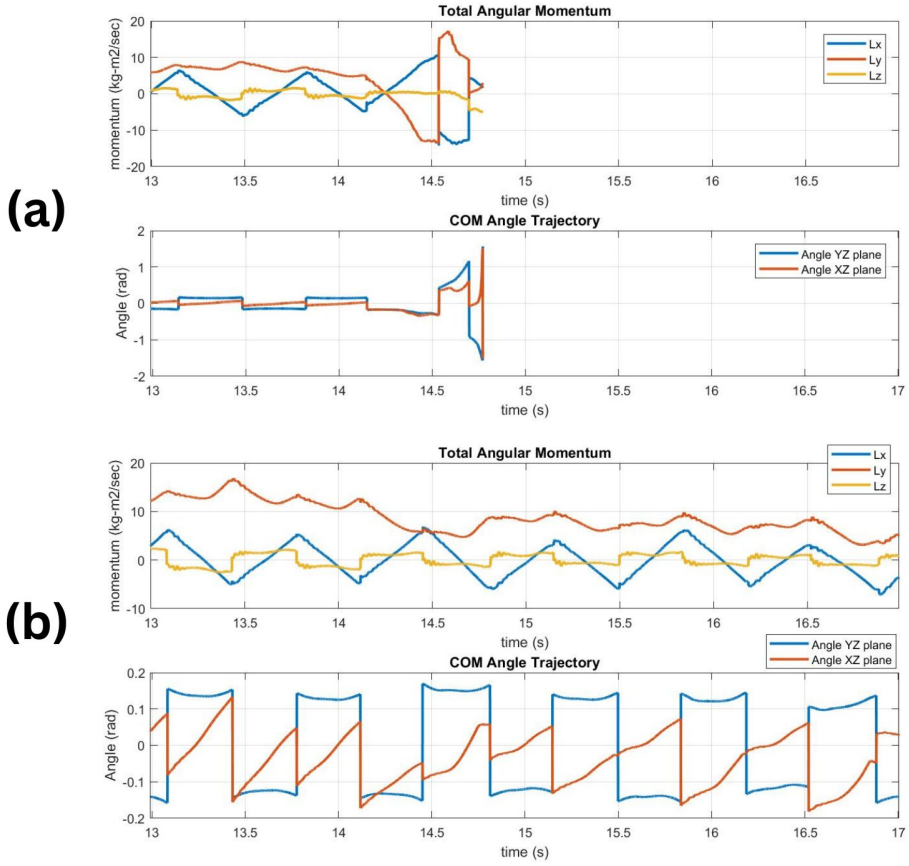


Figure 4.6: Angular momentum and CoM angle over time with and without ankle torque to stabilize walking forward with perturbations at $t = 14$ sec (a) without ankle torque and (b) with ankle torque. Note, that only the relevant time portion of the plot is shown $(13 < t < 17)$ to highlight the effects of the perturbation. In (a), there is no data after ~ 14.7 sec because the simulation fails at this time. Data continues until the end of the simulation for (b) because the robot is able to fully recover after the perturbation.

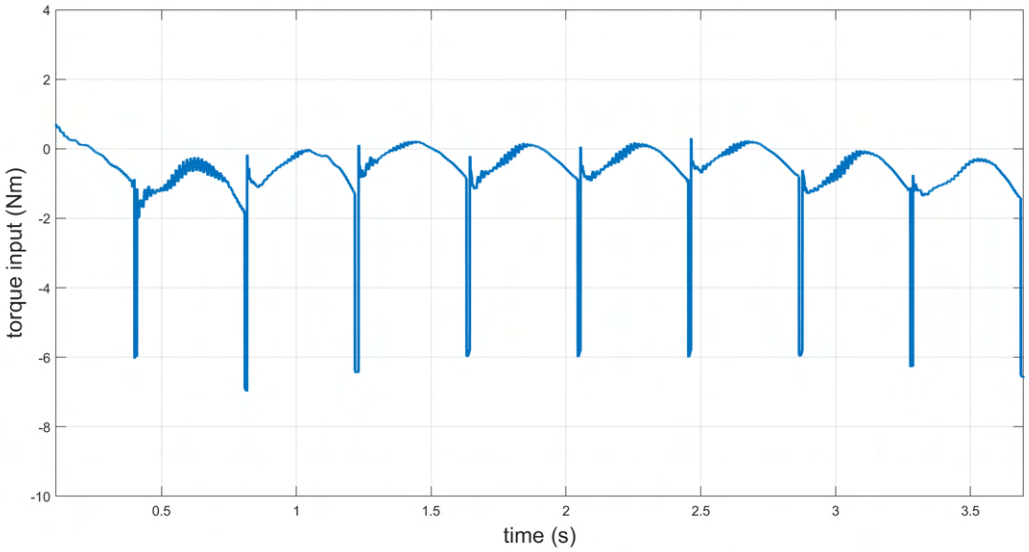


Figure 4.7: Simulated stance ankle torque vs time using MPC for stair climbing.

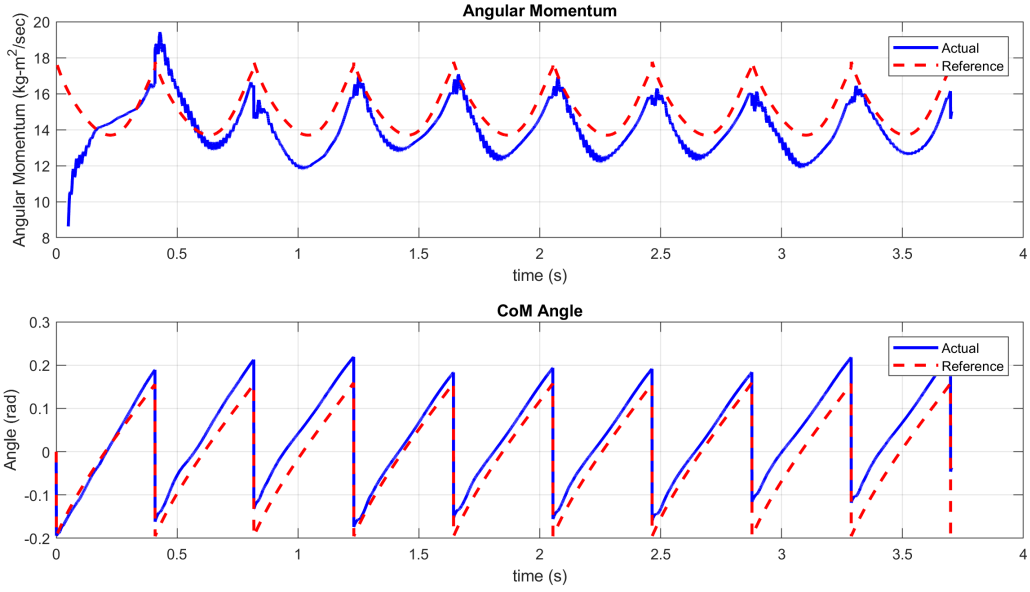


Figure 4.8: Nominal and simulated angular momentum and CoM angle over time using MPC to determine stance ankle torque to stabilize sagittal motion and (lateral) foot placement to stabilize lateral motion during stair climbing.

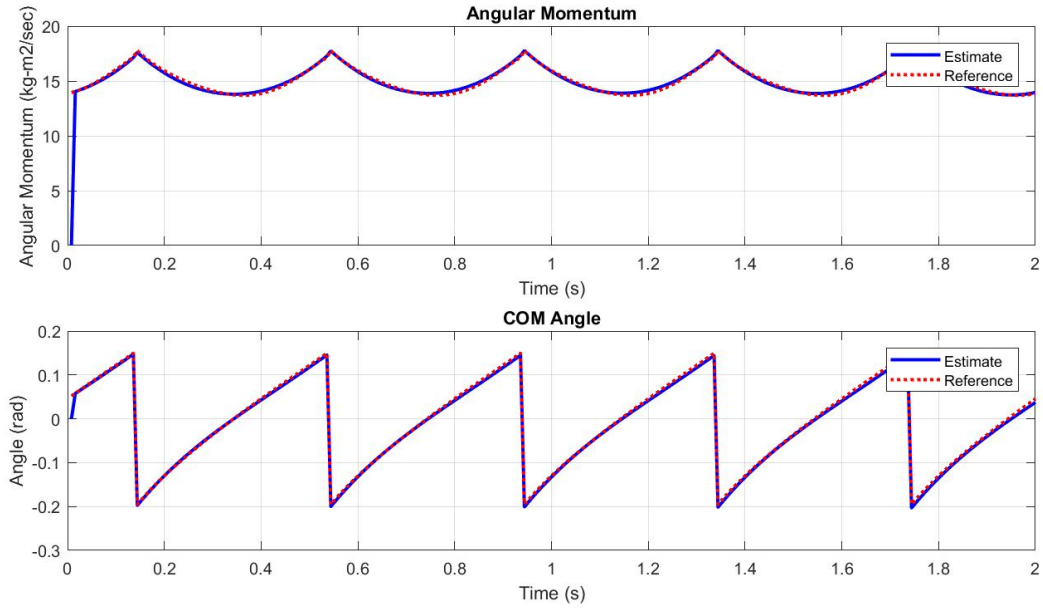


Figure 4.9: CoM angle and angular momentum over time steps for a horizon length $N = 5T$ on the planar nonlinear ALIP model.

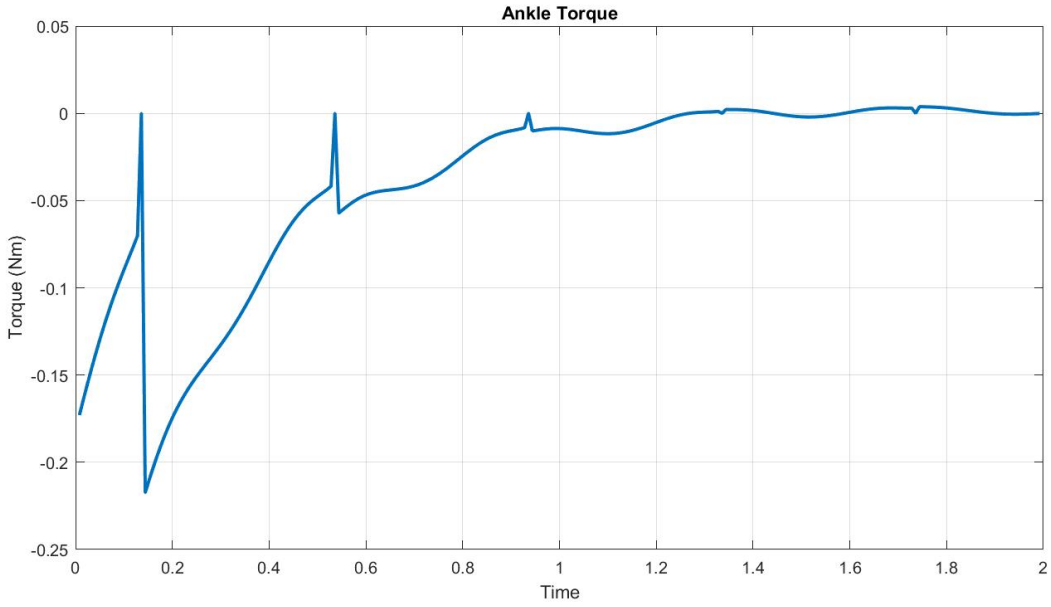


Figure 4.10: Stance ankle torque over a horizon length $N = 5T$ on the planar nonlinear ALIP model.

Chapter 5

Real-world Revelations - Hardware Results and Discussion

This chapter delves into the practical implementation of the controllers outlined in Sections 3.5, 3.6, and 3.7 on the physical 20 DoF Cassie bipedal robot hardware. While the achievements in the SimMechanics simulation environment were commendable, the pivotal challenge lies in bridging the gap between simulation and reality. Roboticists encounter persistent difficulties in translating simulated outcomes to physical hardware due to factors such as motor friction, sensor inaccuracies, and unforeseen variables not accounted for in simulation environments.

5.1 Navigating the sim-to-real Gap for Cassie Hardware

A paramount concern in deploying controllers on Cassie hardware is adhering to the stringent maximum controller execution time of 2 kHz. The passivity-based controller (PBC) detailed in Section 3.5 adeptly operates within this temporal constraint. In contrast, the model predictive controller (MPC) discussed in Section 3.6 demands a higher execution time. To surmount this challenge, we strategically offloaded the MPC computations to a secondary computer. The execution time was further optimized through the utilization of CasADi, an open-source tool for nonlinear optimization and algorithmic differentiation [101, 102]. The MPC computation now runs at an approximate duration of 500 μs. Given that it is unnecessary to compute a desired ankle torque value for every time step, the utilization of a secondary computer for MPC, with a slower execution time dedicated to ankle torque values,

poses no hindrance. This claim is substantiated by the positive outcomes observed in SimMechanics simulation experiments, where the transfer of ankle torque computations to the secondary computer did not result in any noticeable deviations in results.

In addressing the observed spikes in ankle torque values during the SimMechanics simulation, as discussed in Section 4.2, we introduced an impactful approach to mitigate these spikes. Specifically, we developed an impact map based on the linearization about the nominal trajectories. A detailed discussion of this methodology can be found in Appendix D. This enhancement represents a crucial step in refining the control strategy, aiming to minimize unexpected spikes in control output and further enhance the stability and performance of Cassie for hardware experiments.

5.2 Maximum Walking Speed on Various Inclined Surfaces

In evaluating the performance of Cassie equipped with the novel controller structure developed in this thesis, a series of tests were conducted to determine its maximum walking speed on diverse inclinations. Previous walking controllers for Cassie hardware demonstrated speeds of 2 m/s on level ground [31], but exploration of maximum speeds on inclined surfaces was lacking.

With the implementation of our new controller, Cassie showcased an enhanced walking capability, achieving a comfortable speed of up to 2.2 m/s on a flat treadmill [103]. Extending the analysis to inclined surfaces, Cassie maintained a commendable speed of 2 m/s on a treadmill inclined at 4 degrees [104] and 8 degrees [105]. Even on steeper inclinations, such as a 15- and 20-degree slope, Cassie exhibited a noteworthy walking speed, reaching up to 1.5 m/s [106, 4]. These findings underscore the adaptability and improved performance achieved through the integration of the developed controller structure, opening avenues for enhanced mobility on surfaces with varying degrees of incline.

5.3 Continuous Walking on Changing Incline

In our subsequent experiment, we conducted a continuous walking test on a dynamically changing incline. The experiment commenced on a level surface with a 0-degree incline, progressively transitioning to the maximum incline of 20 degrees, and then returning to the initial 0-degree incline, as depicted in Figure 5.1 and Figure 5.2 and whose accompanying video can be found in [3]. Throughout the experiment, the treadmill maintained a constant speed of 0.9 m/s. To streamline the complexity of the experiment, we employed only three nominal trajectories:

- **Nominal Trajectory for Flat Ground Walking:** Implemented during Cassie's movement on inclines ranging from 0 to 7 degrees.
- **Nominal Trajectory for 8-Degree Incline:** Applied as Cassie traversed inclines ranging from 8 to 14 degrees.
- **Nominal Trajectory for 15-Degree Incline:** Utilized when Cassie navigated inclines ranging from 15 to 20 degrees.

The use of nominal trajectories on inclines for which the trajectories were not specifically optimized showcased the robustness of the controller. This experiment demonstrated that perfect trajectories for all situations are not a prerequisite for stability——specifically, a nominal trajectory optimized for a 15-degree incline successfully operated on a 20-degree inclined surface.

Moreover, all nominal trajectories assumed a constant nominal speed of 0.5 m/s, introducing additional disturbance to the system since the treadmill operated at 0.9 m/s during the experiment. This underscored the controller's ability to maintain stability and adjust seamlessly to varying walking speeds, irrespective of the speed specified in the nominal trajectory. The experimental configuration provided a

comprehensive assessment of the developed controller's capacity to adapt and ensure stability during continuous walking on inclines.

Figure 5.3 provides insight into the left and right ankle torque values during this experiment. Importantly, the ankle torque values consistently remain well below the maximum and minimum thresholds of ± 23 N, as allowed by the Quadratic Program (QP) for the MPC.

5.4 Transitioning from Stationary Flat Ground to Inclined Moving Treadmill

In our subsequent experiment, we tasked the Cassie bipedal robot with walking from stationary flat ground to an inclined moving treadmill. This challenging scenario was systematically tested for inclined transitions at 4 degrees [104], 8 degrees [105], 15 degrees [106], and 20 degrees [4]. Figure 5.5 and Figure 5.6 shows outtakes for the experiment of the transition from stationary flat ground to a 20-degree incline. These experiments presented two significant sources of disturbance, both of which our controller adeptly surmounted.

The first challenge involved negotiating the large gaps between the stationary ground and the inclined moving treadmill. This transition demanded a seamless adjustment in gait and stability, which our controller effectively managed. The second challenge arose from the fact that the treadmill was in motion at 0.2 m/s while Cassie initiated the transition from stationary flat ground to the treadmill. Despite this differential in speed, Cassie not only successfully navigated the transition but also exhibited the capability to accommodate increased treadmill speeds.

The controller demonstrated its resilience by overcoming both disturbances encountered during the transition process. Furthermore, we successfully escalated the treadmill speed to the maximum values detailed in Section 5.2 post-transition, affirming the adaptability and robust performance of the developed controller in challenging real-world scenarios.

Figure 5.4 provides a visual representation of the left and right ankle torque values during the experiment, illustrating Cassie's transition from flat stationary ground to the 20-degree inclined moving treadmill. While the ankle torque values approach the saturation threshold of ± 23 N, as allowed by the QP for the MPC, they remain below this limit, underscoring the controller's effectiveness in managing the complexities of dynamic transitions.

5.5 Navigating on and off of a Moving Walkway

In our culminating experiment, we tasked Cassie with traversing on and off a moving walkway——an essential capability for robots in human-centric environments such as airports and shopping malls. The integration of robots into everyday scenarios necessitates their adept navigation through commonplace features like moving walkways, which typically exhibit average speeds ranging from 0.5 to 0.83 m/s [107]. Our objective was to validate Cassie's ability to seamlessly adapt to different speeds on the moving walkway.

We successfully tested Cassie's performance on moving walkways operating at speeds of 0.5 m/s [5], 0.8 m/s [108], and 1.2 m/s [6]. Outtakes from the experiment, as depicted in Figure 5.7 and Figure 5.8, highlight Cassie's agility in walking on and off a 0.5 m/s moving walkway.

A notable challenge introduced during this experiment was the presence of a gantry, which did not extend far enough to cover the entire length of the treadmill used to simulate the moving walkway. As a result, Cassie experienced significant tugging forces from the gantry. During the initial setup, when Cassie was walked to the end of the treadmill, it stumbled and nearly fell due to the tugging forces but swiftly recovered. Furthermore, despite operator error leading to Cassie approaching the moving walkway at a diagonal for the 0.8 m/s and 1.2 m/s scenarios, Cassie adeptly navigated onto and off the moving walkway. A remarkable video showcasing

Cassie's successful navigation on and off a 1.2 m/s moving walkway, despite stumbling from the tugging gantry (shown in Figure 5.9 and Figure 5.10) and approaching at a diagonal, was featured in the IEEE Spectrum Video Friday edition on November 24, 2023. This demonstration underscores the robustness and adaptability of our controller in addressing real-world challenges during robot locomotion on dynamic surfaces.

Figure 5.11 provides insight into the left and right ankle torque values during Cassie's traversal on and off a 1.2 m/s moving treadmill. Noteworthy spikes are evident between the 290-second and 295-second marks, corresponding to the moment when Cassie stumbled. Importantly, the ankle torque values consistently remain well below the maximum and minimum thresholds of ± 23 N, as allowed by the QP for the MPC. This observation attests to the controller's ability to manage unexpected disturbances while maintaining stability, further validating its effectiveness in real-world scenarios.

5.6 Discussion and Conclusion

This chapter details the implementation of controllers on the physical 20 DoF Cassie bipedal robot hardware, moving beyond commendable SimMechanics simulation outcomes. The transition to physical hardware accentuates the persistent challenge of the sim-to-real gap, encompassing issues like motor friction, sensor inaccuracies, and unforeseen variables absent in simulation environments.

The stringent maximum controller execution time of 2 kHz on Cassie hardware poses a critical concern. While the passivity-based controller (PBC) successfully adheres to this constraint, the model predictive controller (MPC) demands a longer execution time. Our strategic approach involves offloading MPC computations to a secondary computer, optimizing execution time with CasADi. Successful simulation experiments affirm the viability of this offloading strategy.

Evaluating Cassie's performance with the developed controller structure reveals enhanced walking capabilities. Cassie achieves speeds of up to 2.2 m/s on a flat treadmill, with maintained speeds of 2 m/s on an 8-degree inclined treadmill and 1.5 m/s on a 20-degree incline. This demonstrates the controller's adaptability and improved performance across varying incline levels.

A continuous walking experiment on changing incline further validates the robustness of the controller. Nominal trajectories, not perfectly optimized for all situations, showcase the controller's stability during continuous walking on inclines, emphasizing its adaptability.

Transitions from stationary flat ground to an inclined moving treadmill present unique challenges, including negotiating gaps and adapting to treadmill motion. The controller adeptly manages these disturbances, enabling successful transitions, even at increased treadmill speeds. This resilience underscores the adaptability and robust performance of the developed controller.

Cassie's navigation on and off a moving walkway, despite challenges introduced by a tugging gantry and operator error, highlights the controller's robustness and adaptability in real-world scenarios. This successful demonstration contributes valuable insights for integrating bipedal robots into dynamic human-centric environments.

In conclusion, this chapter demonstrates successful implementation and real-world applicability of controllers on Cassie hardware. By addressing challenges in execution time, hardware constraints, and adapting to dynamic surfaces, our controller contributes to the advancement of bipedal robots, fostering capabilities for enhanced mobility in diverse environments.

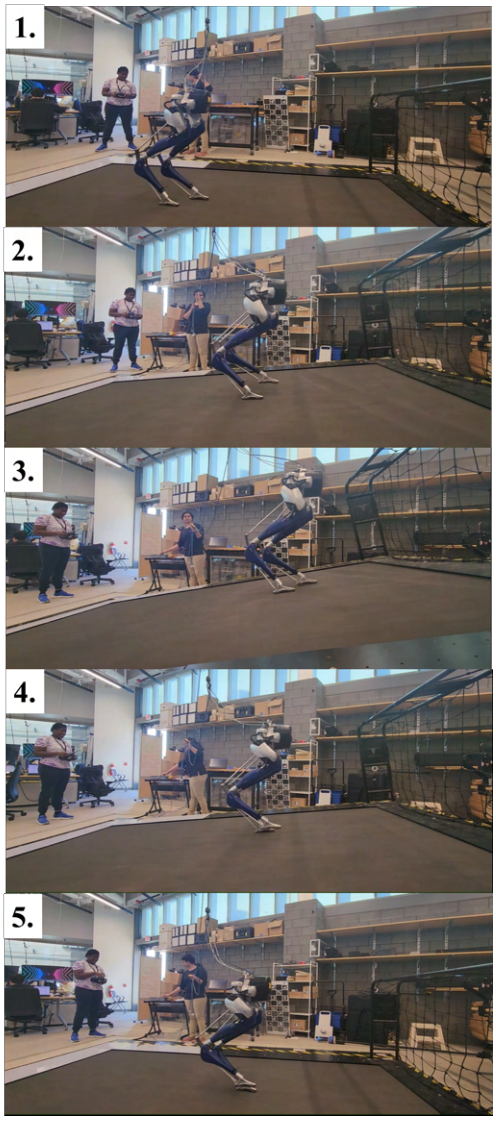


Figure 5.1: A series of images depicting hardware results of the Cassie bipedal robot walking on an inclined moving treadmill moving at a constant speed of 0.9 m/s from a side view perspective. The treadmill is gradually inclined from 0 degrees to its maximum incline of 20 degrees and back to 0 degrees. The original video can be found in [3].

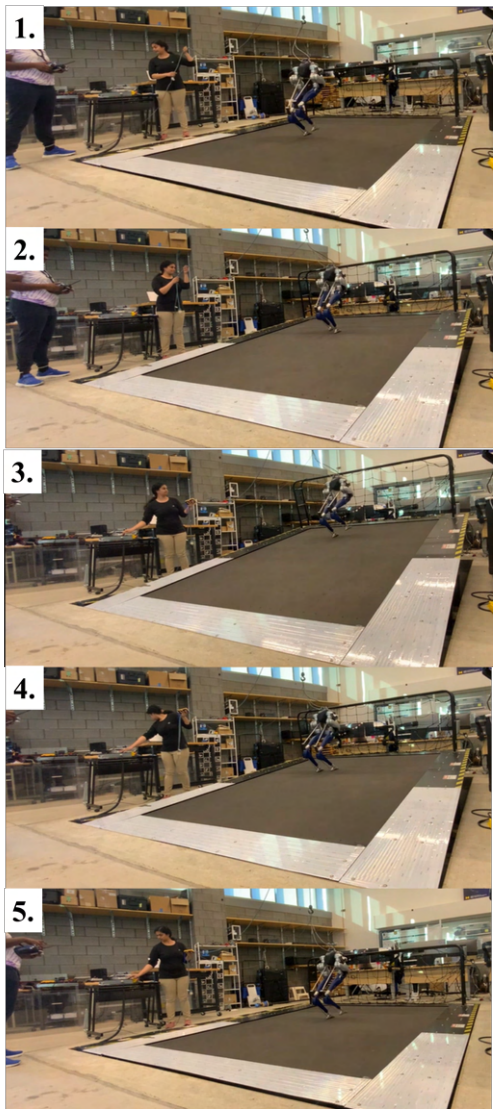


Figure 5.2: A series of images depicting hardware results of the Cassie bipedal robot walking on an inclined moving treadmill moving at a constant speed of 0.9 m/s from a back view perspective. The treadmill is gradually inclined from 0 degrees to its maximum incline of 20 degrees and back to 0 degrees. The original video can be found in [3].

Figure 5.3: Plot of left and right ankle torque values for hardware experiment on the Cassie bipedal robot walking on an inclined moving treadmill moving at a constant speed of 0.9 m/s. The treadmill is gradually inclined from 0 degrees to its maximum incline of 20 degrees and back to 0 degrees.

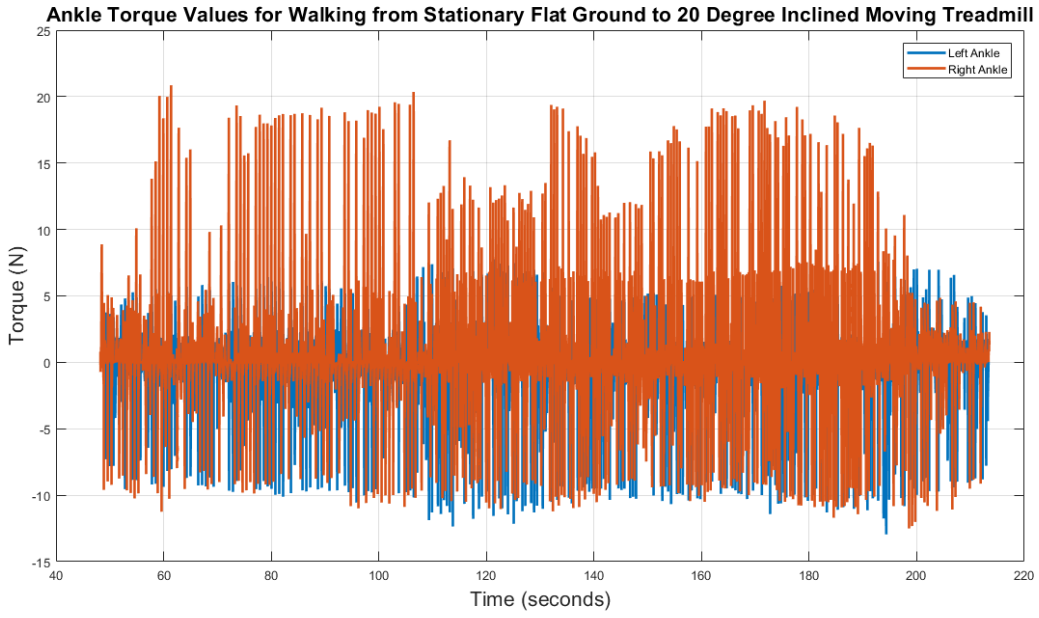


Figure 5.4: Plot of left and right ankle torque values for hardware experiment on the Cassie bipedal robot walking from stationary flat ground to a 0.2 m/s moving treadmill inclined at 20 degrees. Once Cassie steps on the treadmill, the treadmill's speed is gradually increased to 1.5 m/s.

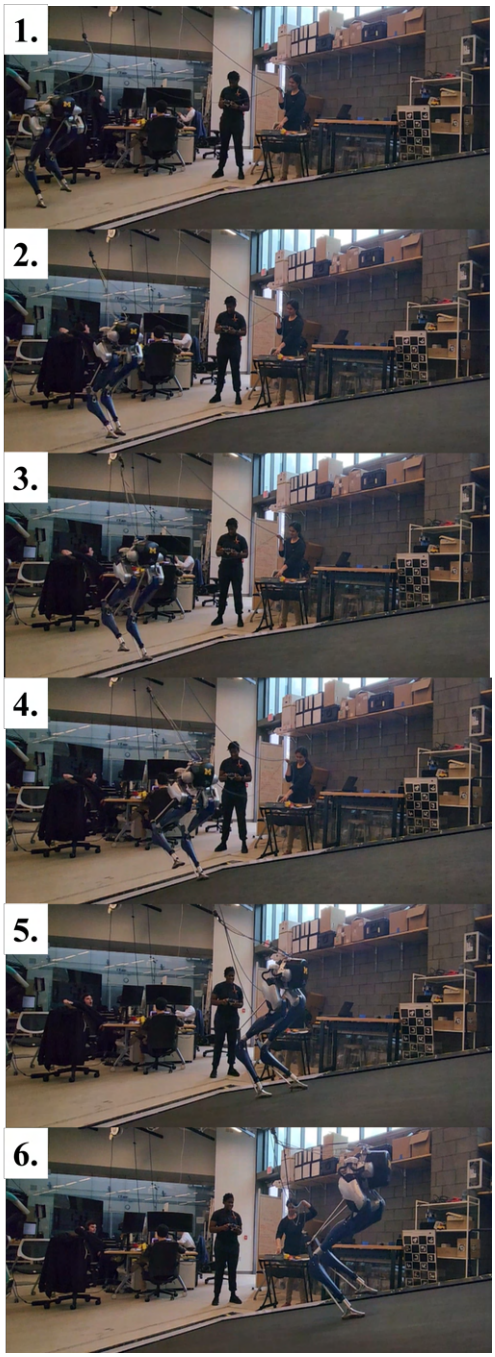


Figure 5.5: A series of images depicting hardware results of the Cassie bipedal robot walking from flat, stationary ground to a 0.2 m/s moving treadmill inclined at 20 degrees from a side view perspective. Once Cassie steps on the treadmill, the treadmill's speed is gradually increased to 1.5 m/s. The original video can be found in [4].

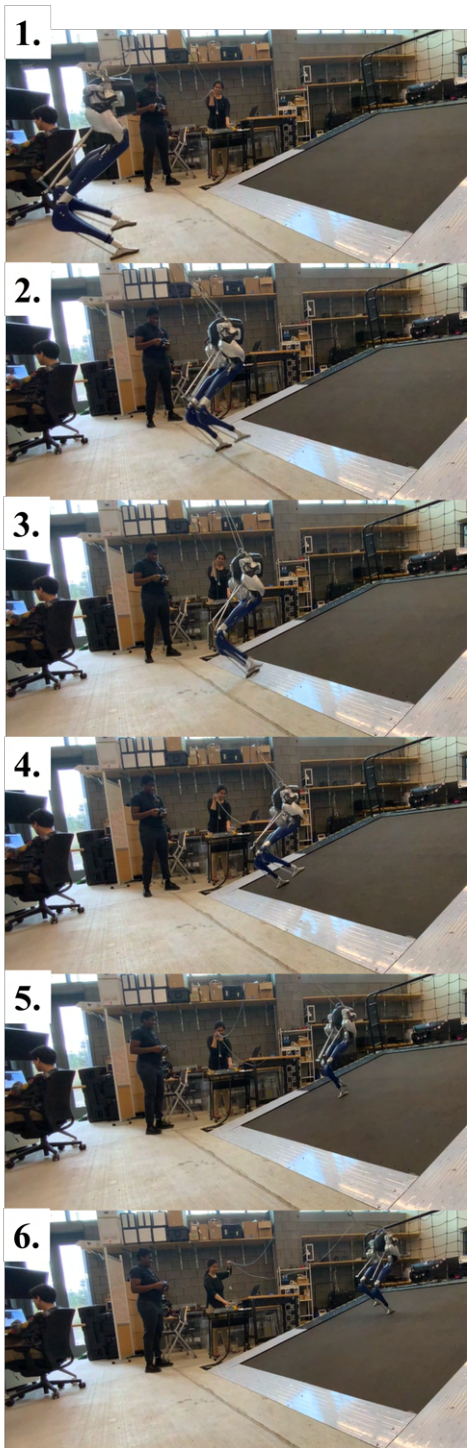


Figure 5.6: A series of images depicting hardware results of the Cassie bipedal robot walking from flat, stationary ground to a 0.2 m/s moving treadmill inclined at 20 degrees from a back view perspective. Once Cassie steps on the treadmill, the treadmill's speed is gradually increased to 1.5 m/s. The original video can be found in [4].

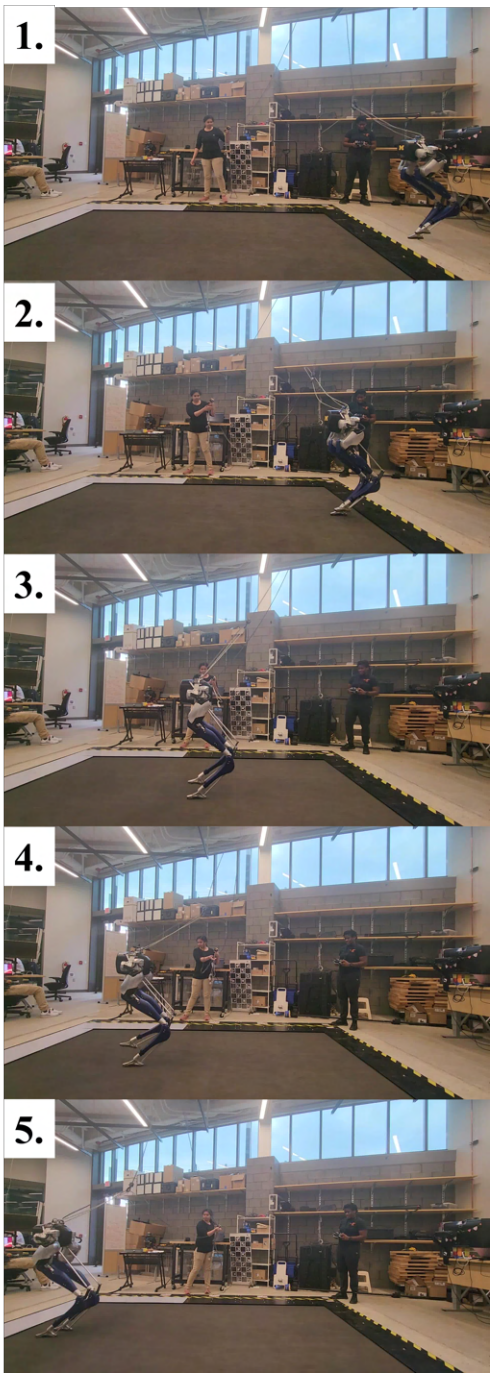


Figure 5.7: A series of images depicting hardware results of the Cassie bipedal robot walking on to and off of a 0.5 m/s moving walkway (treadmill) from a side view perspective. The original video can be found in [5].

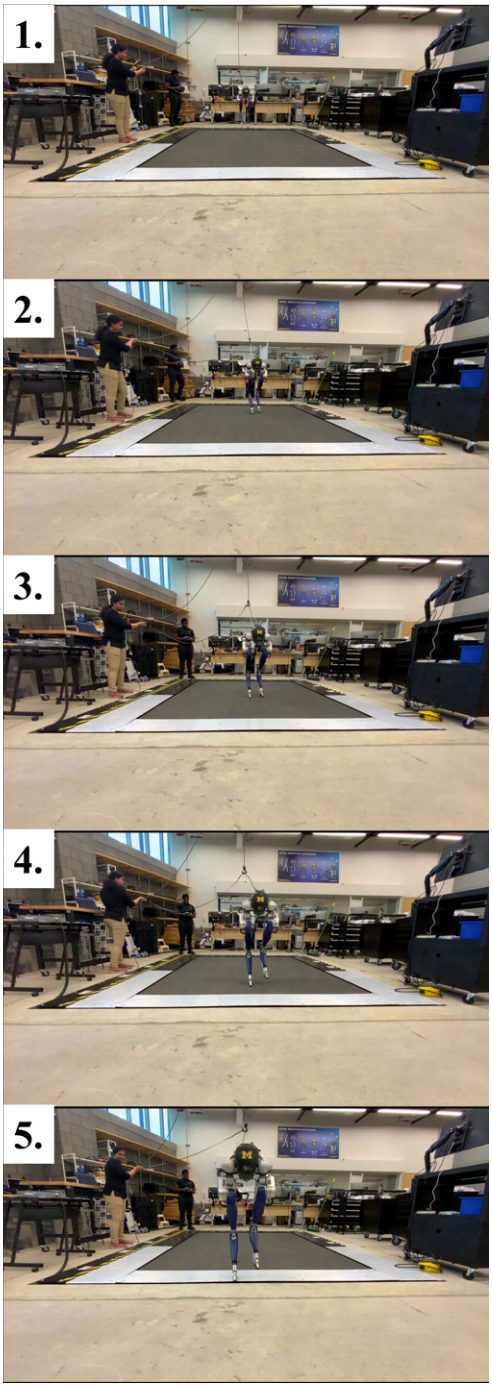


Figure 5.8: A series of images depicting hardware results of the Cassie bipedal robot walking on to and off of a 0.5 m/s moving walkway (treadmill) from a front view perspective. The original video can be found in [5].

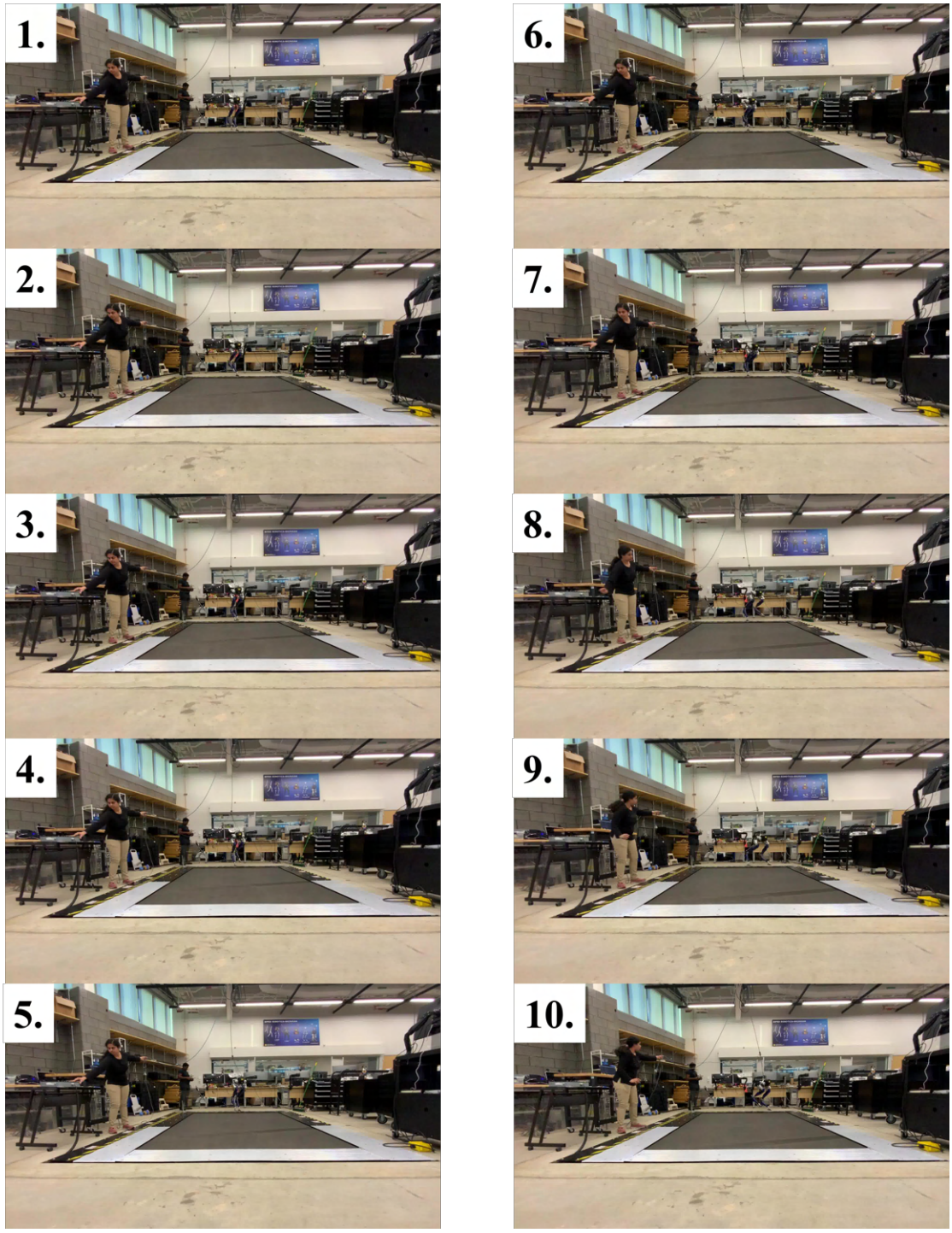


Figure 5.9: Part 1 of a series of images depicting Cassie stumbling during the setup for the moving walkway experiment. Part 2 of the images is shown in Figure 5.10. The original video can be found in [6].

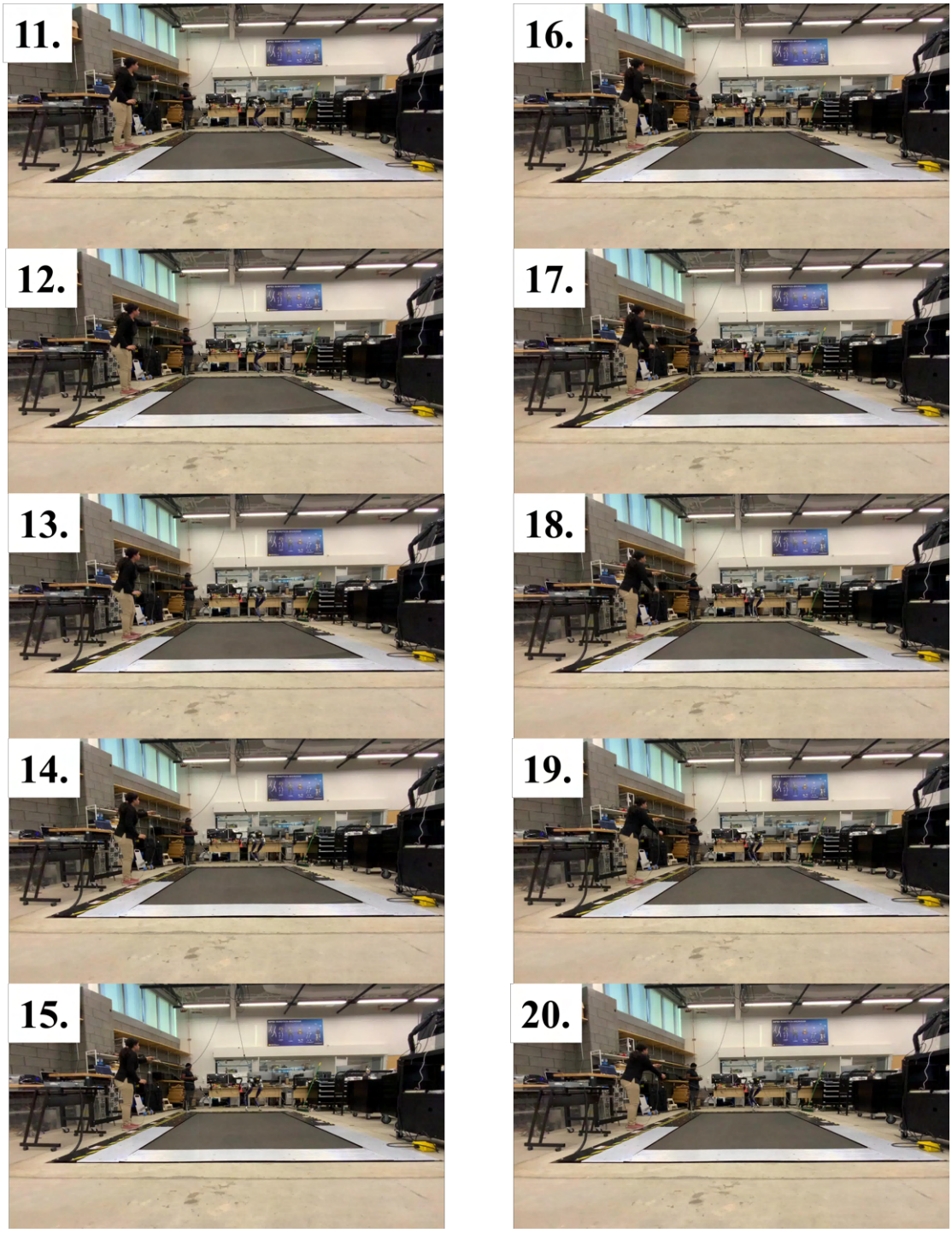


Figure 5.10: Part 2 of a series of images depicting Cassie stumbling during the setup for the moving walkway experiment. Part 1 of the images is shown in Figure 5.9. The original video can be found in [6].

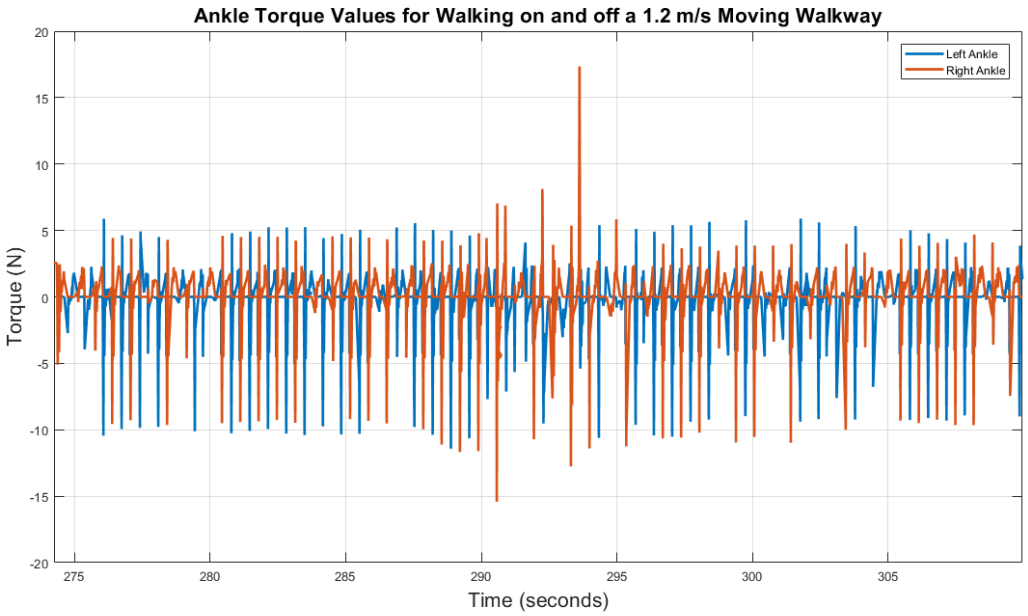


Figure 5.11: Plot of left and right ankle torque values for hardware experiment on the Cassie bipedal robot walking on to and off of a 1.2 m/s moving walkway (treadmill).

Chapter 6

Future Work and Conclusions

The research developed in this thesis makes significant strides in addressing the challenges of stair climbing for underactuated bipedal robots. Nevertheless, additional avenues for future exploration and improvement emerge from this work. In this chapter, we outline the next steps for future research and development to further enhance the capabilities and practical applications of the proposed stair-climbing algorithm.

6.1 Implementing Stair Climbing on Cassie Hardware

A pivotal step in advancing Cassie's mobility involves extending its capabilities to navigate stairs. This future work focuses on implementing stair climbing and testing the algorithm's performance across various stair geometries. The goal is to adapt the existing walking controllers to seamlessly negotiate stairs with varying step heights, tread depths, and incline angles. Fine-tuning gait parameters specific to stair traversal, assessing stability and robustness, and enabling real-time adaptation to changes in stair geometry are essential components of this research. Additionally, exploring the integration of sensory feedback mechanisms, such as depth cameras or proximity sensors, will enhance Cassie's ability to perceive and respond to the stair environment. This comprehensive approach aims to broaden Cassie's mobility spectrum, making it adept at navigating complex terrains in real-world scenarios.

6.2 Advancing Stair Negotiation – Walking Downstairs

Expanding Cassie's mobility to include walking downstairs is a natural progression in enhancing its versatility. However, due to Cassie's kinematics, addressing this challenge necessitates a phased approach.

6.2.1 Phase 1: Backward-Facing Descent

The initial step involves mastering walking downstairs with Cassie facing backward. This approach mitigates the risk of Cassie's legs colliding with the stairs during descent, ensuring a safe and controlled traversal. Implementing algorithms that enable Cassie to descend stairs in a backward-facing orientation requires careful consideration of gait parameters, stability adjustments, and real-time adaptation to stair geometry changes. This phase aims to establish a foundation for safe stair negotiation before progressing to more complex scenarios.

6.2.2 Phase 2: Forward-Facing Descent

Once the backward-facing descent has been successfully implemented and optimized, the subsequent phase involves developing algorithms for Cassie to walk downstairs while facing forward. This represents a more intricate challenge due to the potential risk of leg-stair collisions. Addressing this requires innovative solutions in gait planning, trajectory optimization, and dynamic stability adjustments. Ensuring Cassie's ability to descend stairs in a forward-facing orientation will significantly broaden its applicability, enabling more intuitive navigation in environments with varying terrains.

6.2.3 Kinematic Considerations

Throughout both phases, careful attention must be given to Cassie's kinematics. Analyzing joint configurations, limb trajectories, and ensuring adequate leg clearance

during descent are crucial aspects. Iterative refinement of the algorithms based on empirical testing and simulation results will be paramount to achieving a seamless and reliable stair descent capability.

6.2.4 Integration with Sensory Feedback

Similar to the stair-climbing future work, integrating sensory feedback mechanisms will enhance Cassie's perception during stair descent. Depth cameras, proximity sensors, or other relevant sensors can provide real-time information about the stairs, aiding in adjusting Cassie's gait and ensuring a safe and efficient descent.

In conclusion, advancing Cassie's stair negotiation capabilities to include walking downstairs represents a significant future work. By initially mastering backward-facing descent and subsequently progressing to forward-facing descent, this research aims to augment Cassie's mobility, making it proficient in navigating a broader spectrum of real-world environments with stairs.

6.3 Outdoor Navigation for Cassie

To broaden Cassie's mobility for outdoor environments, the focus is on adapting existing algorithms, particularly for stair climbing, to handle natural slopes, uneven surfaces, and various outdoor stairs. This involves fine-tuning the stair-climbing algorithm for seamless navigation in challenging outdoor conditions.

Following successful lab testing, the next step is to conduct targeted outdoor experiments. Locations like the Wave Field and the steep, uneven grassy slope at the University of Michigan provide diverse terrains for validating Cassie's performance in real-world scenarios.

However, outdoor experiments with Cassie come with logistical challenges and safety considerations. Operating an out-of-warranty Cassie biped outdoors demands careful planning, risk assessment, and contingency measures to ensure both the safety

of Cassie and the success of the experiments.

In summary, future work on outdoor navigation for Cassie centers on adapting stair-climbing algorithms, conducting targeted outdoor experiments, addressing logistical challenges, and ensuring safety. This research aims to position Cassie as a capable and versatile bipedal robot for navigating the complexities of outdoor environments.

6.4 Generalizing Control Strategy for Different Bipedal Robots

The next phase of this research involves assessing the transferability of the current algorithm from the Cassie robot to other underactuated bipedal platforms. Validating the control strategy on diverse robots, such as Agility Robotics' Digit bipedal robot, will provide insights into its versatility and potential for broader applications. This extension aims to demonstrate the adaptability of the developed control algorithm across various hardware architectures, advancing its utility in diverse robotic scenarios.

6.5 Conclusions

In conclusion, the development and exploration of the Angular Linear Inverted Pendulum (ALIP) model, particularly its adaptation for stair climbing, mark a significant advancement in the control paradigm for bipedal robots. The traditional limitations associated with variations in center of mass height and non-kinematic restrictions on foot placement have been addressed through the innovative integration of smooth trajectory-following for the virtual pendulum. The devised control strategy, combining virtual constraint-based control and a model predictive control algorithm, has demonstrated success in stabilizing a stair-climbing gait that transcends reliance solely on foot placement.

Simulations on a detailed model of the Cassie biped and corresponding hardware experiments illustrate the algorithm's efficacy, showcasing its potential to enhance

the robustness of walking gaits on inclined surfaces. This research not only provides a practical solution to the challenges posed by stair climbing but also lays the groundwork for further advancements in bipedal robot mobility.

Looking ahead, future research directions should focus on refining and optimizing the proposed algorithm to address specific challenges in real-world scenarios. Additionally, further exploration of underactuated bipedal robots and their applications in diverse industries holds great promise. By continuing to unlock the full potential of these robots, we can pave the way for their widespread utilization and integration into various fields, ultimately contributing to advancements in automation, exploration, and assistance in real-world applications. The stair-climbing algorithm presented in this thesis thus stands as a crucial stepping stone toward the realization of more capable and versatile bipedal robots.

Appendix A

Using Lagrangian Mechanics to derive the Robot Equations on a Point Mass System

In this appendix, we use Lagrangian Mechanics to derive the robot equations for the three-link bipedal robot depicted in Figure A.1.

The Lagrangian is defined by

$$L = KE - PE \tag{A.1}$$

where KE is kinetic energy and PE is potential energy. Kinetic energy for a point mass can be calculated as follows:

$$KE = \frac{1}{2}m({v_x}^2 + {v_y}^2) \tag{A.2}$$

where m is mass, v_x is the velocity in the horizontal direction, and v_y is the velocity in the vertical direction. Potential energy for a point mass can be calculated as follows:

$$PE = mgh \tag{A.3}$$

where m is mass, g is gravity, and h is height.

Taking the partial derivative of Equation (A.1) gets the Lagrange Equation in a more usable form:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \Gamma \tag{A.4}$$

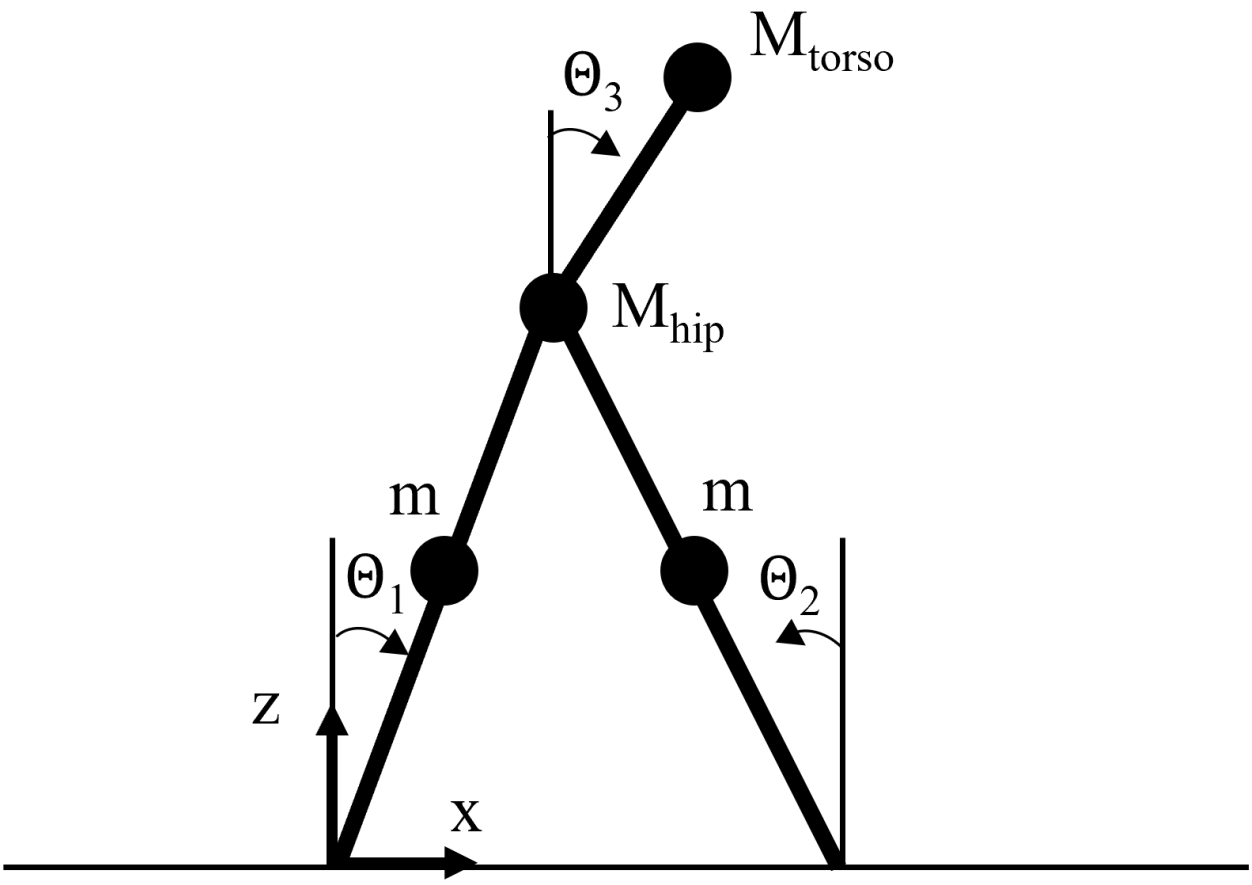


Figure A.1: Schematic of the three-link walker in Matlab.

where Γ is the external and internal forces, q is the generalized coordinates and $\dot{q}$ is the generalized velocities.

Equation (A.4) is known as the Euler-Lagrangian. This equation can be rearranged into a different form to make mathematical manipulation easier. This form is:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \Gamma = Bu \tag{A.5}$$

where D is the mass inertial matrix, C is the centrifugal and coriolis forces matrix, G is the gravity vector, B is the input matrix, and u is the torques of the motors. (A.5) are the **equations of motion**.

Generating Equation (A.5) is straightforward. Approximating the mass of each link of a robot as a single point mass, we define the variable p_i as the position vector for the mass of link i. That is,

$$p_i = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = f(q) \tag{A.6}$$

where p_x is the horizontal coordinates and p_y is the vertical coordinates. Velocity is found by taking the derivative:

$$v_i = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \frac{\partial f}{\partial q}\dot{q} \tag{A.7}$$

where v_x is the velocity in the horizontal direction and v_y is the velocity in the vertical direction.

Using Equation (A.2), the kinetic energy for each mass is derived as

$$KE_i = \frac{1}{2} m (v_i)^T (v_i) \tag{A.8}$$

$$= \frac{1}{2} m \dot{q}^T \left(\frac{\partial f}{\partial q}\right)^T \left(\frac{\partial f}{\partial q}\right) \dot{q} \tag{A.9}$$

$$= \frac{1}{2} \dot{q}^T D(q) \dot{q} \tag{A.10}$$

where $D(q) = \left[m \left(\frac{\partial f}{\partial q}\right)^T \left(\frac{\partial f}{\partial q}\right) \right]$ and is a symmetric matrix since it is the matrix multiplication of a matrix and its transpose.

The total kinetic energy of the robot would simply be the sum of the kinetic energies of each of the masses,

$$KE = \sum_{i=1}^{4} \frac{1}{2} \dot{q}^T \underbrace{\left[m \left(\frac{\partial f}{\partial q}\right)^T \left(\frac{\partial f}{\partial q}\right) \right]}_{D_i(q)} \dot{q} \tag{A.11}$$

Plugging in Equation (A.1) into Equation (A.4) yields

$$\frac{d}{dt} \frac{\partial (KE - PE)}{\partial \dot{q}} - \frac{\partial (KE - PE)}{\partial q} = \Gamma \tag{A.12}$$

Taking the partial derivative of KE from Equation (A.10), yields

$$\frac{d}{dt} \left[\frac{\partial}{\partial \dot{q}} \big(\frac{1}{2} \dot{q}^T D(q) \dot{q} \big) - \cancelto{0}{\frac{\partial [PE]}{\partial \dot{q}}} \right] - \frac{\partial}{\partial q} \left[\frac{1}{2} \dot{q}^T D(q) \dot{q} - PE \right] = \Gamma \tag{A.13}$$

$$\frac{d}{dt} \left[\frac{1}{\cancel{2}} (\cancel{2}) D(q) \dot{q} - 0 \right] - \frac{\partial}{\partial q} \left[\frac{1}{2} \dot{q}^T D(q) \dot{q} - PE \right] = \Gamma \tag{A.14}$$

$$\left[D(q) \ddot{q} + \underbrace{\frac{\partial}{\partial q} \big[D(q) \dot{q} \big] \dot{q}}_{=C(q,\dot{q})\dot{q}} \right] - \left[\underbrace{\frac{1}{2} \frac{\partial}{\partial q} \big[\dot{q}^T D(q) \dot{q} \big]}_{=C(q,\dot{q})\dot{q}} - \underbrace{\frac{\partial [PE]}{\partial q}}_{=G(q)} \right] = \Gamma \tag{A.15}$$

Thus, we have derived the equations of motion of Equation (A.5).

Remark: The C matrix is typically derived in terms of Christoffel symbols [2].

Remark: It is worth reiterating that for practical applications, we do not typically approximate robot systems as point masses. Nevertheless, understanding Lagrangian mechanics on a point mass system can better help conceptualize the math hidden behind software programs used to derive the robot equations for complex systems like Cassie. This fundamental knowledge of bipedal robots (and more) are discussed in the Biped Bootcamp technical document that this author wrote. See Appendix F.

Appendix B

Input-Output Feedback Linearization

Input-Output Feedback Linearization is a common approach to control nonlinear systems such as bipedal robots [2, 109, 110]. Though the practical applications are limited due to requiring an accurate model of the system, its relatively simple nature makes it an ideal controller to use in simulation. In practice, we use a passivity-based controller that uses nonlinear input-output feedback [82].

The output function is defined as

$$y(x) = h_0(q) - h_d(q, {p_{sw}^x}^{des}, {p_{sw}^y}^{des}, {p_{sw}^z}^{des}, t) \tag{B.1}$$

where h_0 is the collection of virtual constraints and h_d provides the desired trajectories for the virtual constraints. In part due to precedent [61, 35, 1] and in part due to the new ALIP model of Section 2.3.2 that is being used for this paper, the

virtual constraints are defined as follows:

$$h_0(q) = \begin{bmatrix} \text{absolute torso pitch} \\ \text{absolute torso roll} \\ \text{stance hip yaw} \\ \text{swing hip yaw} \\ \text{pendulum length} \\ p^x_{st\to sw} \\ p^y_{st\to sw} \\ p^z_{st\to sw} \\ \text{absolute swing toe pitch} \end{bmatrix} \tag{B.2}$$

where the pendulum length describes the vector r_c from the stance foot to the CoM and $p_{st\to sw}$ is the vector emanating from the stance foot and ending at the swing foot.

We design an input-output linearizing controller such that

$$\ddot{y} + K_D\dot{y} + K_P y = 0 \tag{B.3}$$

where K_D and K_P are user-defined derivative and proportional controller gains, respectively. When designing the controller, we check that the decoupling matrix is full rank and we assume that the ankle torque is known. The required value of the ankle torque is developed in Section 3.6.

The first and second derivatives of y are defined as

$$\begin{aligned} \dot{y} &= -\dot{h}_d + Jh\dot{q} \\ \ddot{y} &= -\ddot{h}_d + Jh\ddot{q} + \delta Jh\dot{q} \end{aligned} \tag{B.4}$$

where Jh is the jacobian of the constraints h at a given time step and δJh is the

partial derivative of the jacobian of the constraints at given time step with respect to q.

Plugging in Equation (B.4) into Equation (B.3) yields

$$-\ddot{h}_d + Jh\ddot{q} + \delta Jh\dot{q} + K_D(-\dot{h}_d + Jh\dot{q}) + K_P h = 0 \tag{B.5}$$

where $\ddot{q}$ is known from the continuous dynamics defined in Equation (2.5). That is,

$$\begin{aligned} \ddot{q} &= M([1:20],:)u_9 - b[1:20] \\ &= \overline{M}u_9 - \overline{b} \end{aligned} \tag{B.6}$$

where $\overline{M}$ and $\overline{b}$ describe the first 20 rows of the matrices M and b, which are defined as

$$\begin{aligned} M &= \begin{bmatrix} D & -J_{st}^T & -J_s^T \\ J_{st} & 0 & 0 \\ J_s & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} B_9 \\ 0 \\ 0 \end{bmatrix} \\ b &= \begin{bmatrix} D & -J_{st}^T & -J_s^T \\ J_{st} & 0 & 0 \\ J_s & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} C\dot{q} + G - B_1 u_1 \\ \dot{J}_{st}\dot{q} \\ \dot{J}_s\dot{q} \end{bmatrix}. \end{aligned} \tag{B.7}$$

Thus, plugging in Equation (B.6) into Equation (B.5) and solving for the control input u_9, the final controller is defined as

$$u_9 = [Jh\overline{M}]^{-1}[\ddot{h}_d + Jh\overline{b} - \delta Jh\dot{q} - K_D(-\dot{h}_d + Jh\dot{q}) - K_P h]. \tag{B.8}$$

Appendix C

A Naïve Approach for Control Computation in Model Predictive Control

This appendix discusses a naïve approach to solving the Model Predictive Control problem discussed in Section 3.6. The discrete-time model formulation and predictive steps are the same as discussed in Section 3.6. The change thus comes in the control computation.

For $N > 2$, 3.24 is an underdetermined system of linear equations in the unknown vector of control decisions, u_k^{seq}. We propose to determine a value for it via

$$u_k^{\text{seq}\ *} := \underset{\Gamma_k u_k^{\text{seq}} = \left(x_{k+N}^{\text{des}} - S_k x_k\right)}{\arg\min} ||u_k^{\text{seq}}||^2. \tag{C.1}$$

And then, in the spirit of MPC, we only apply the first component, which we call u_k^*.

Using standard results, we note that

$$u_k^{\text{seq}\ *} = \Gamma_k^T \cdot (\Gamma_k \cdot \Gamma_k^T)^{-1}(x_{k+N}^{\text{des}} - S_k x_k). \tag{C.2}$$

Therefore, we have

$$u_k^* = u_k^{\text{seq}\ *}(1),$$

the first component of $u_k^{\text{seq}\ *}$.

Remark: This is a naïve approach to implementing an MPC problem because

we impose a single equality constraint at the end of the control horizon and do not impose bounds on u_k when solving the least squares problem. For a more robust implementation, a QP with inequality constraints poses a less naïve optimization problem than Equation (C.1).

Appendix D

Developing an Impact Map by Linearizing over a Nominal Trajectory

In this appendix, we develop an impact map based off of a linearization about a nominal trajectory. This was used to reduce the spikes in ankle torque computed by the MPC controller witnessed in the SimMechanics simulation experiments discussed in Section 4.2.

We utilize the Taylor series to linearize about a nominal trajectory. Linearization about a nominal trajectory using a Taylor series is a common technique in control theory and system analysis. The idea is to approximate a nonlinear system by a linear one in the vicinity of a nominal trajectory or operating point.

Consider a nonlinear function $f(x)$. We can approximate $f(x)$ about a nominal trajectory $\bar{x}$ using a Taylor series expansion:

$$f(x) = f(\bar{x}) + \left.\frac{df}{dx}\right|_{x=\bar{x}} (x-\bar{x}) + \frac{1}{2}\left.\frac{d^2f}{dx^2}\right|_{x=\bar{x}} (x-\bar{x})^2 + \frac{1}{6}\left.\frac{d^3f}{dx^3}\right|_{x=\bar{x}} (x-\bar{x})^3 ... \tag{D.1}$$

which can be written as

$$f(x) = f(\bar{x}) + \left.\frac{df}{dx}\right|_{x=\bar{x}} (x-\bar{x}) + \text{higher order terms.} \tag{D.2}$$

For x sufficiently close to $\bar{x}$, the higher order terms will be very close to zero.

Thus, we can drop these terms to obtain the approximation

$$f(x) \approx f(\bar{x}) + \left.\frac{df}{dx}\right|_{x=\bar{x}} (x - \bar{x}). \tag{D.3}$$

Let

$$\delta x = x - \bar{x} \tag{D.4}$$

Thus,

$$x = \bar{x} + \delta x. \tag{D.5}$$

Using Equation (D.5), Equation (D.3) becomes

$$f(\bar{x} + \delta x) \approx f(\bar{x}) + \left.\frac{df}{dx}\right|_{x=\bar{x}} \delta x, \tag{D.6}$$

where $\bar{x}$ is the nominal trajectory, δx is the error between the actual value and nominal trajectory, and $\frac{df}{dx}\big|_{x=\bar{x}}$ is the derivative of the function f evaluated at the nominal trajectory $\bar{x}$.

We now develop an impact map based off a linearization about the nominal trajectory for center of mass (CoM) angle, $\bar{\theta}_c^+$, and angular momentum. We start with CoM angle.

D.1 CoM Angle

Figure D.1 depicts the relevant variables for computing the impact map. Note that $\bar{\theta}_c^+$ refers to the CoM angle after impact and $\bar{\theta}_c^-$ refers to CoM angle before impact.

From geometry, we have

$$r_c^- \cos(\bar{\theta}_c^-) = r_c^+ \cos(\bar{\theta}_c^+) + h. \tag{D.7}$$

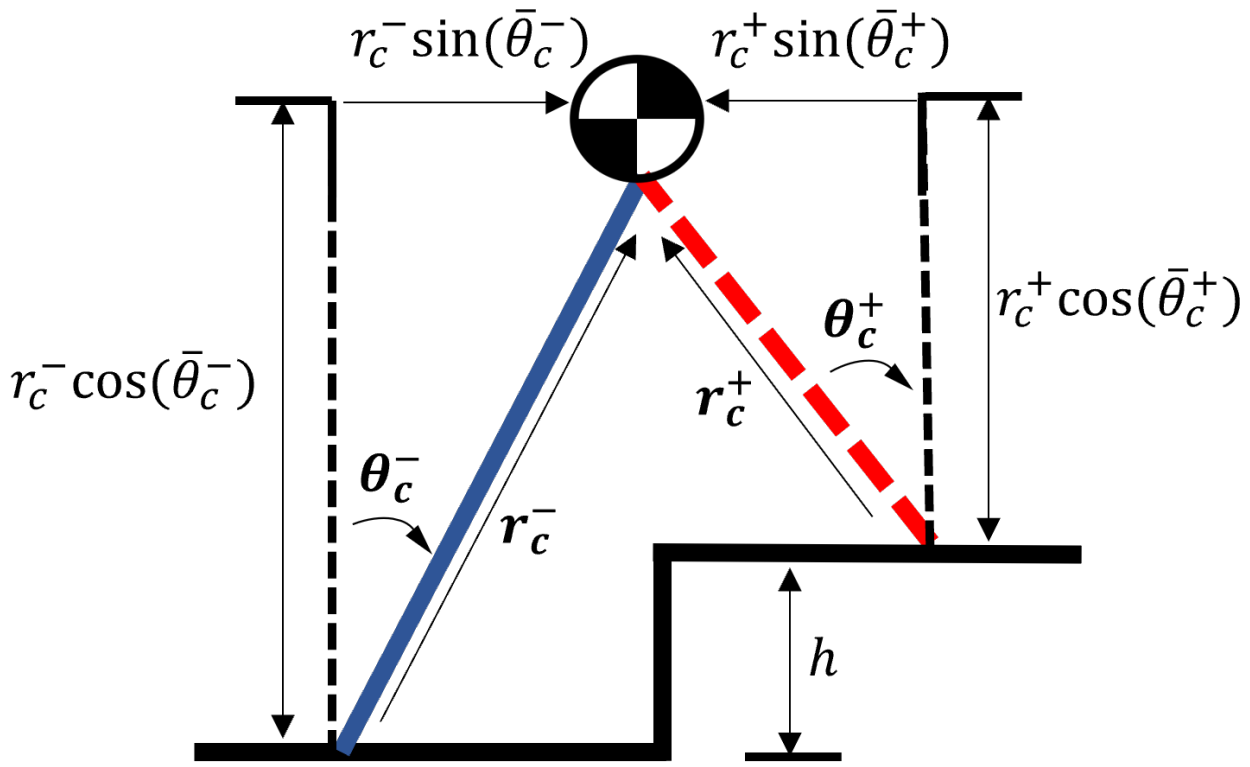


Figure D.1: Schematic depicting geometry variables for computing impact map based off of a lineariziation of a nominal trajectory.

While we wish to perfectly follow the nominal trajectory $\bar{\theta}_c^+$, in reality, we do not. Instead, we have an error δx. Thus, Equation (D.7) becomes

$$r_c^- \cos(\bar{\theta}_c^- + \delta x) = r_c^+ \cos(\bar{\theta}_c^+ + \delta x) + h. \tag{D.8}$$

Applying linearization on $\cos(\bar{\theta}_c^- + \delta x)$ and $\cos(\bar{\theta}_c^+ + \delta x)$ using Equation (D.6) on Equation (D.8) yields

$$r_c^- \left[\cos(\bar{\theta}_c^-) + \frac{d}{d\theta_c^-} \cos(\theta_c^-)\Big|_{\theta_c^- = \bar{\theta}_c^-} \right] = r_c^+ \left[\cos(\bar{\theta}_c^+) + \frac{d}{d\theta_c^+} \cos(\theta_c^+)\Big|_{\theta_c^+ = \bar{\theta}_c^+} \right] + h \tag{D.9}$$

$$r_c^- \left[\cos(\bar{\theta}_c^-) - \sin(\bar{\theta}_c^-)\delta\theta_c^- \right] = r_c^+ \left[\cos(\bar{\theta}_c^+) - \sin(\bar{\theta}_c^+)\delta\theta_c^+ \right] + h \tag{D.10}$$

$$r_c^- \cos(\bar{\theta}_c^-) - r_c^- \sin(\bar{\theta}_c^-)\delta\theta_c^- = r_c^+ \cos(\bar{\theta}_c^+) - r_c^+ \sin(\bar{\theta}_c^+)\delta\theta_c^+ + h \tag{D.11}$$

Plugging in Equation (D.7) into Equation (D.11) yields

$$\cancel{r_c^+ \cos(\bar{\theta}_c^+)} + \cancel{h} - r_c^- \sin(\bar{\theta}_c^-)\delta\theta_c^- = \cancel{r_c^+ \cos(\bar{\theta}_c^+)} - r_c^+ \sin(\bar{\theta}_c^+)\delta\theta_c^+ + \cancel{h} \tag{D.12}$$

$$-r_c^- \sin(\bar{\theta}_c^-)\delta\theta_c^- = -r_c^+ \sin(\bar{\theta}_c^+)\delta\theta_c^+. \tag{D.13}$$

Solving for the error $\delta\theta_c^+$ yields,

$$\delta\theta_c^+ = \frac{r_c^- \sin(\bar{\theta}_c^-)}{r_c^- \sin(\bar{\theta}_c^+)} \delta\theta_c^- \tag{D.14}$$

where $r_c^- \sin(\bar{\theta}_c^+)$ is a constant given by the nominal trajectory.

To find the actual value of θ_c^+, we use Equation (D.5) applied to the CoM angle after impact θ_c^+. That is,

$$\theta_c^+ = \delta\theta_c^+ + \bar{\theta}_c^+. \tag{D.15}$$

Plugging in Equation (D.14) into Equation (D.15) yields

$$\theta_c^+ = \frac{r_c^- \sin(\bar{\theta}_c^-)}{r_c^- \sin(\bar{\theta}_c^+)} \delta\theta_c^- + \bar{\theta}_c^+. \tag{D.16}$$

D.2 Angular Momentum

We now compute the impact map for angular momentum by linearizing about its nominal trajectory. We start by using the impact map derived for angular momentum in Section 3.7.2. From Equation (3.31), we know that if we were perfectly following the nominal trajectory, the angular momentum after impact would be

$$\begin{aligned} \bar{L}_A^+ = \bar{L}_B^- + m \Bigg[& P^z_{\text{st} \to \text{sw}} \Big(r_c^- \cos(\bar{\theta}_c^-) \dot{\bar{\theta}}_c^- + \dot{r}_c^- \sin(\bar{\theta}_c^-) \Big) - \\ & P^x_{\text{st} \to \text{sw}} \Big(- r_c^- \sin(\bar{\theta}_c^-) \dot{\bar{\theta}}_c^- + \dot{r}_c^- \cos(\bar{\theta}_c^-) \Big) \Bigg]. \end{aligned} \tag{D.17}$$

In reality, we have error δL and $\delta\theta_c$. Thus, Equation (D.17) becomes

$$\begin{aligned} (\bar{L}_A^+ + \delta L^+) = (\bar{L}_B^- + \delta L^-) + m \Bigg[& P^z_{\text{st} \to \text{sw}} \Big(r_c^- \cos(\bar{\theta}_c^- + \delta\theta_c^-)(\dot{\bar{\theta}}_c^- + \delta\dot{\theta}_c) \\ & + \dot{r}_c^- \sin(\bar{\theta}_c^- + \delta\theta_c^-) \Big) - P^x_{\text{st} \to \text{sw}} \Big(- r_c^- \sin(\bar{\theta}_c^- + \delta\theta_c^-)(\dot{\bar{\theta}}_c^- + \delta\dot{\bar{\theta}}_c^-) \\ & + \dot{r}_c^- \cos(\bar{\theta}_c^- + \delta\theta_c^-) \Big) \Bigg]. \end{aligned} \tag{D.18}$$

We now linearize Equation (D.18) using Equation (D.3).

For clarity, we do the computations in segments.

$$\bar{L}^+ + \delta L^+ \approx \bar{L}^+ + \left(\frac{\partial(\bar{L}^+)}{\partial L^+} + \frac{\partial(\bar{L}^+)}{\partial \theta_c^+} \right) \cdot \delta L^+ \tag{D.19}$$

$$= \bar{L}^+ + (1) \delta L^+ \tag{D.20}$$

$$= \bar{L}^+ + \delta L^+ \tag{D.21}$$

$$\bar{L}^- + \delta L^- \approx \bar{L}^- + \left(\frac{\partial(\bar{L}^-)}{\partial L^-} + \frac{\partial(\bar{L}^-)}{\partial \theta_c^-} \right) \cdot \delta L^- \tag{D.22}$$

$$= \bar{L}^- + (1)\delta L^- \tag{D.23}$$

$$= \bar{L}^- + \delta L^- \tag{D.24}$$

$$\cos(\bar{\theta}_c^- + \delta\theta_c^-) \approx \cos(\bar{\theta}_c^-) + \left(\frac{\partial\big(\cos(\bar{\theta}_c^-)\big)}{\partial L^-} + \frac{\partial\big(\cos(\bar{\theta}_c^-)\big)}{\partial \theta_c^-} \right) \cdot \delta\theta_c^- \tag{D.25}$$

$$= \cos(\bar{\theta}_c^-) - \big(\sin(\bar{\theta}_c^-)\big) \cdot \delta\theta_c^- \tag{D.26}$$

$$= \cos(\bar{\theta}_c^-) - \delta\theta_c^- \sin(\bar{\theta}_c^-) \tag{D.27}$$

$$\dot{\bar{\theta}}_c^- + \delta\dot{\theta}_c^- \approx \dot{\bar{\theta}}_c^- + \left(\frac{\partial(\dot{\bar{\theta}}_c^-)}{\partial L^-} + \frac{\partial(\dot{\bar{\theta}}_c^-)}{\partial \theta_c^-} \right) \cdot \theta_c^- \tag{D.28}$$

$$= \dot{\bar{\theta}}_c^- + \cancel{(0)\delta\theta_c^-} \tag{D.29}$$

$$= \dot{\bar{\theta}}_c^- \tag{D.30}$$

$$\sin(\bar{\theta}_c^- + \delta\theta_c^-) \approx \sin(\bar{\theta}_c^-) + \left(\frac{\partial\big(\sin(\bar{\theta}_c^-)\big)}{\partial L^-} + \frac{\partial\big(\sin(\bar{\theta}_c^-)\big)}{\partial \theta_c^-} \right) \cdot \delta\theta_c^- \tag{D.31}$$

$$= \sin(\bar{\theta}_c^-) + \big(\cos(\bar{\theta}_c^-)\big) \cdot \delta\theta_c^- \tag{D.32}$$

$$= \sin(\bar{\theta}_c^-) + \delta\theta_c^- \cos(\bar{\theta}_c^-) \tag{D.33}$$

Using Equation (D.21), Equation (D.24), Equation (D.27), Equation (D.30), and

Equation (D.33), Equation (D.18) becomes

$$(\bar{L}_A^+ + \delta L^+) = (\bar{L}_B^- + \delta L^-) + m\Bigg[P^z_{\text{st}\,\to\,\text{sw}}\Big(r_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-)$$
$$+\dot{r}_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big]\Big) - P^x_{\text{st}\,\to\,\text{sw}}\Big(-r_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-) \tag{D.34}$$
$$+\dot{r}_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big]\Big)\Bigg].$$

Solving for δL^+ yields,

$$\delta L^+ = \bar{L}_B^- + \delta L^- - \bar{L}_A^+ + m\Bigg[P^z_{\text{st}\,\to\,\text{sw}}\Big(r_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-)$$
$$+\dot{r}_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big]\Big) - P^x_{\text{st}\,\to\,\text{sw}}\Big(-r_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-) \tag{D.35}$$
$$+\dot{r}_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big]\Big)\Bigg].$$

To find the actual value of L^+, we use Equation (D.5) applied to the angular momentum after impact L^+. That is,

$$L^+ = \delta L^+ + \bar{L}^+. \tag{D.36}$$

Plugging in Equation (D.35) into Equation (D.36) yields

$$L^+ = \bar{L}_B^- + \delta L^- - \bar{L}_A^+ + m\Bigg[P^z_{\text{st}\,\to\,\text{sw}}\Big(r_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-)$$
$$+\dot{r}_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big]\Big) - P^x_{\text{st}\,\to\,\text{sw}}\Big(-r_c^-\Big[\sin(\bar{\theta}_c^-) + \delta\theta_c^-\cos(\bar{\theta}_c^-)\Big](\dot{\bar{\theta}}_c^-) \tag{D.37}$$
$$+\dot{r}_c^-\Big[\cos(\bar{\theta}_c^-) - \delta\theta_c^-\sin(\bar{\theta}_c^-)\Big]\Big)\Bigg] + \bar{L}^+.$$

Equation (D.16) and Equation (D.37) form our impact map.

Appendix E

Results in the Ideal Simulator

This section discusses the implementation of the controllers from Appendix B, Section 3.7, and Appendix C on the 20 DoF simulation model of the Cassie robot using Matlab and Simulink in an "ideal simulator". This simulator does not include Cassie's stiff springs in the full order model. It also does not include any state estimation, a necessary feature for hardware implementation. This ideal simulator serves as a proof-of-concept implementation of our stair-climbing controller.

Figure E.1 shows the Cassie robot in the simulated environment on stairs. Note the direction of the positive $x-$ and $z-$axes, which means that a positive rotation about the $y-$axis is rotating down the stairs. This is an important observation for later.

E.1 Initial Tests

As a sanity check, the input-output linearizing controller of Section B was implemented without any control in the sagittal and frontal planes. In other words, the controllers from Appendix C and Section 3.7 were not activated. As expected, the closed-loop system was unstable. The robot would take two steps and then fall backward, shown in Figure E.2.

We then activated the Simplified MPC controller, which resulted in the robot taking 7 steps before falling sideways. We then activated the controller for the

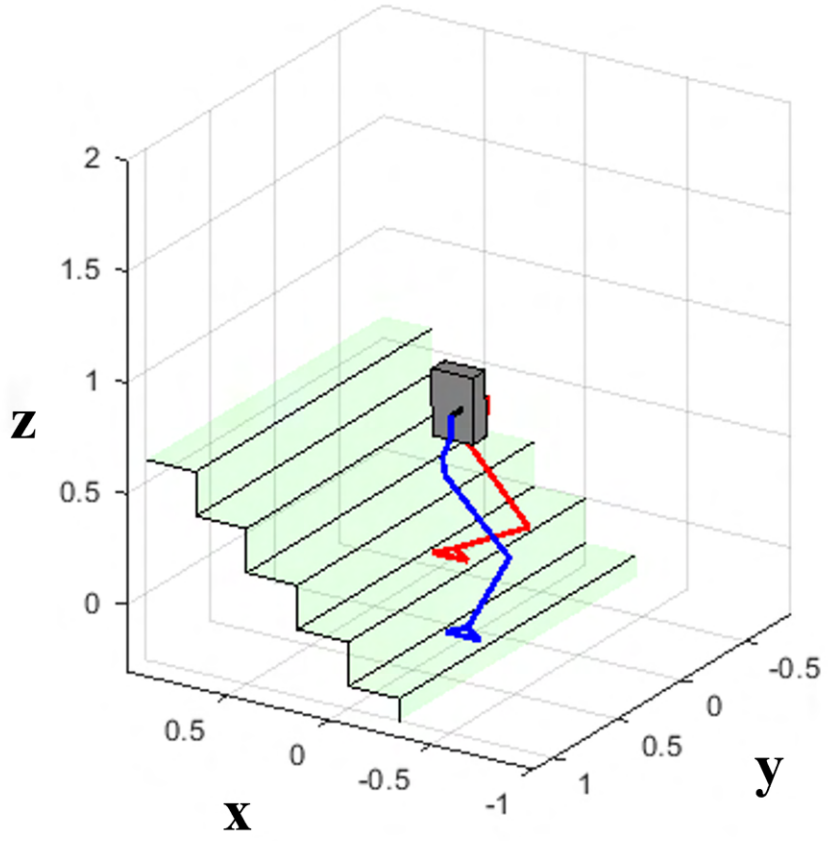


Figure E.1: Simulated 3D model of the Cassie robot using Matlab and Simulink.

lateral dynamics, resulting in the robot being able to walk an unbounded number of steps.

Making the control horizon shorter ($T/2$ instead of T) does not change the robot's ability to walk an unbounded number of steps; however, it does cause the stance ankle torque to saturate more. Extending the horizon to $2T$ not only caused the stance ankle torque to saturate more, but also caused the simulation to fail after just one step.

E.2 Discussion of Results

We will focus discussion on the Simplified MPC approach to ankle torque control, since that is where we made our major contribution.

Even while taking a naïve approach, implementing MPC to determine the stance ankle torque needed to achieve desired angular momentum and CoM angle over time proved successful. Figure E.3 shows outtakes of the animation from the working simulation.

Figure E.4 shows the stance ankle torque inputs calculated via the simplified MPC approach throughout the simulation period of five seconds. A stance ankle torque limit of ± 17 Nm was used. Note the mostly negative values for the stance ankle torques. Without implementing either controller, the robot would fall backward, which results in a positive rotation about the $y-$axis. Therefore, in order to prevent the robot from falling backward, a negative torque must be applied about the $y-$axis, hence the negative stance ankle torque values.

The motors do saturate toward the end of each step, likely as a response to the increase in angular momentum that occurs at the end of each step. Because the input-output linearizing controller is "cancelling" the stance ankle torque, saturation of the stance ankle torque does not prevent the input-output linearizing controller from driving the virtual constraints to zero. If we had been employing the stance

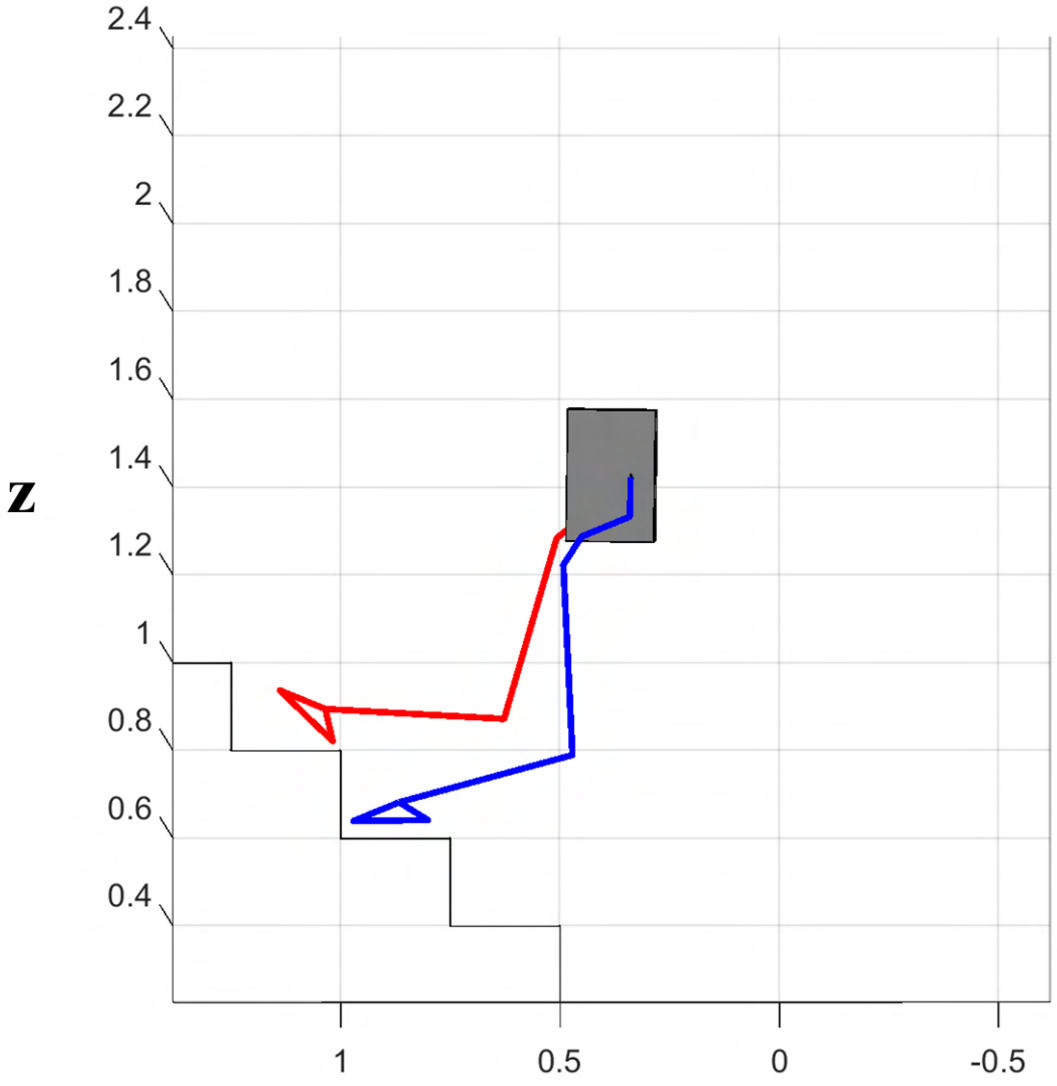


Figure E.2: Cassie falling backward as a result of not adding stabilizing sagittal or lateral controllers.

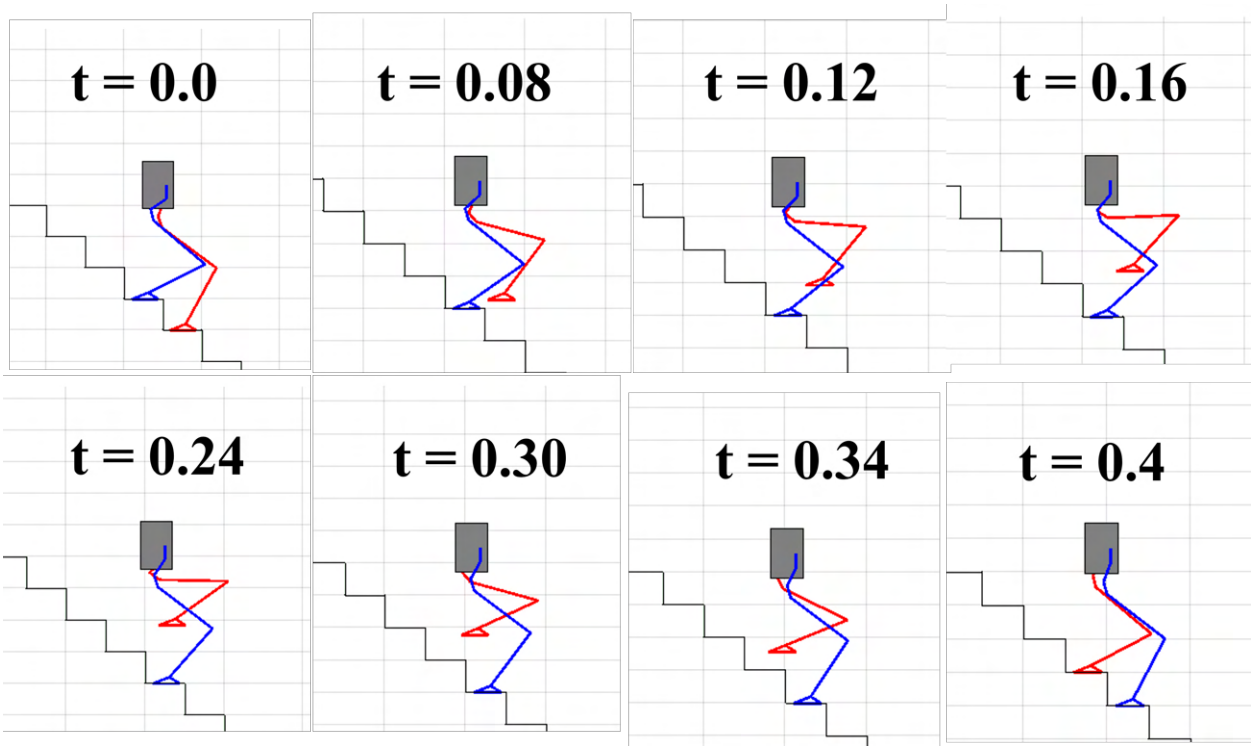


Figure E.3: One-step animation result of stair climbing using MPC to determine stance torque input and ALIP to determine lateral foot placement. The time units are seconds.

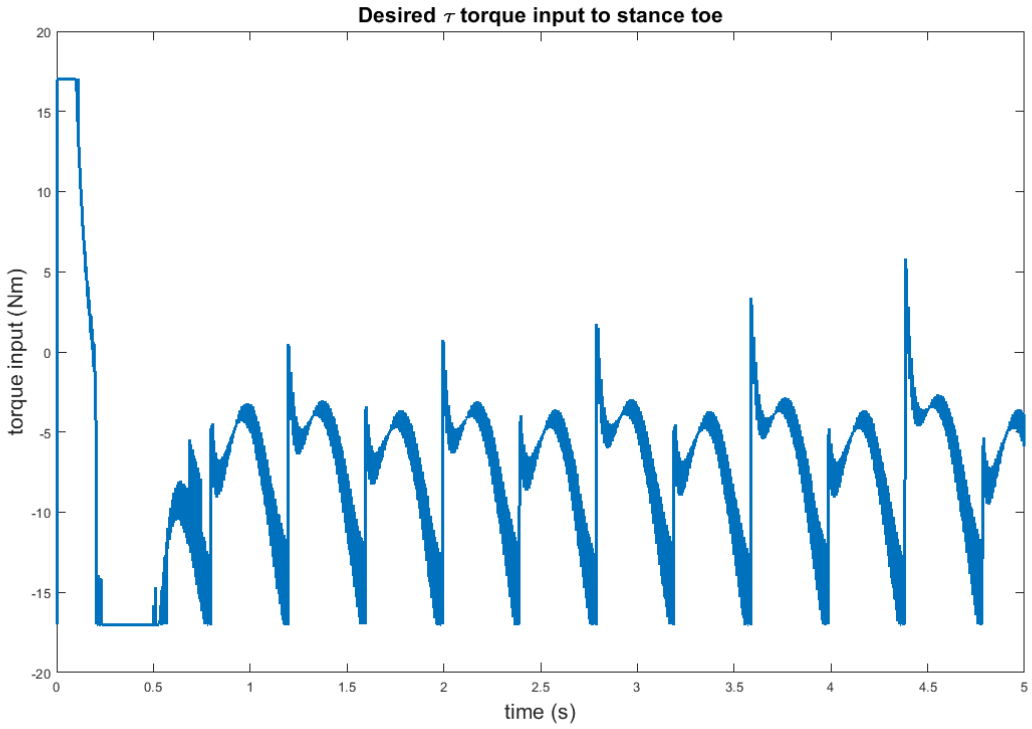


Figure E.4: Stance ankle torque input over time using simplified MPC to determine desired values for the stance ankle torque input.

ankle torque to impose ten virtual constraints, then saturation of the ankle torque could have prevented the controller from maintaining the virtual constraints near zero, possibly resulting in falling of the robot.

Figure E.5 shows the angular momentum and CoM angle over the entire simulation run time of five seconds during which the simulated robot completed 12 steps. The plots shows both the nominal trajectory that was used to set the desired values for the MPC when determining stance ankle torque, as well as the actual simulated values.

Note that even though the simulated trajectory is not exactly following the nominal trajectory, it is still able to achieve a stable periodic orbit. This led us to believe that our controllers would be robust enough to reject small perturbations. Indeed, even when adding a push to the system at the beginning of the simulation

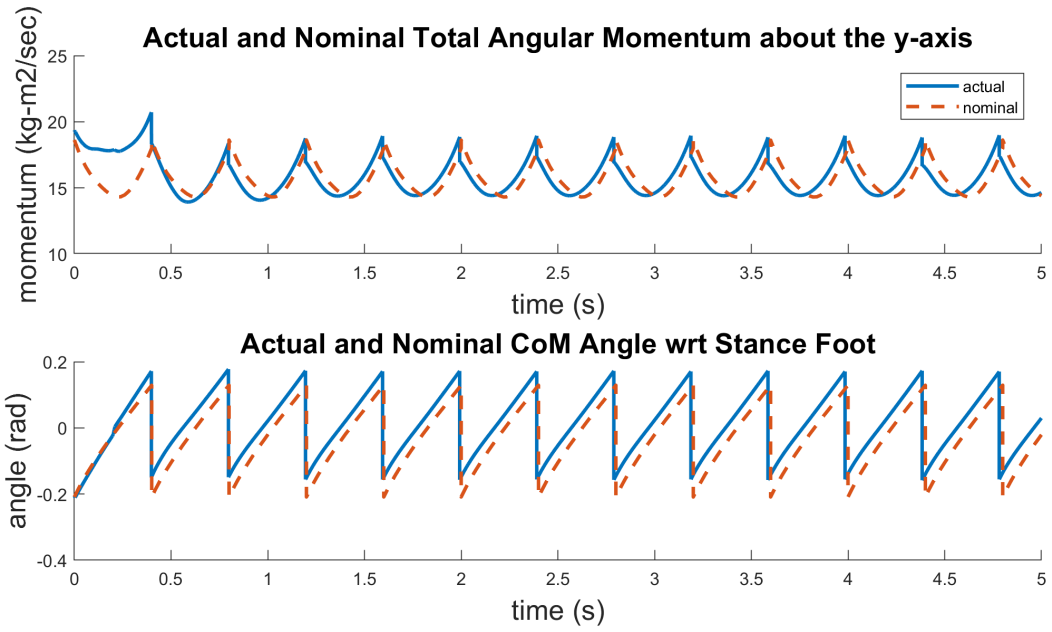


Figure E.5: Nominal and simulated angular momentum and CoM angle over time using MPC to determine stance ankle torque to stabilize sagittal motion, and (lateral) foot placement to stabilize lateral motion.

that resulted in an additional 5.09 kg-m^2/sec of angular momentum (20% increase) being introduced to the system, the controllers were able to adequately recover by the end of the first step and continue on a periodic orbit as shown in Fig. E.6.

We also attempted to perturb the system by increasing the mass value used in the MPC calculation from Cassie's nominal mass of 32 kg to 35 kg. This resulted in the system failing within half of a second with the robot falling forward. Trials with mass set to 34 kg and 33 kg were also tested, but with similar results. We are confident that upgrading from the simplified MPC approach used in this paper to a more standard MPC approach will lead to a more robust controller, capable of handling varying mass values.

To further test the robustness of the controller, we tried varying the stair height, without re-computing the nominal trajectories, to see how the controller will hold up with sub-optimal desired trajectories. The robot was able to take 5 steps before falling when the stair height was changed from the nominal height of 20 cm to

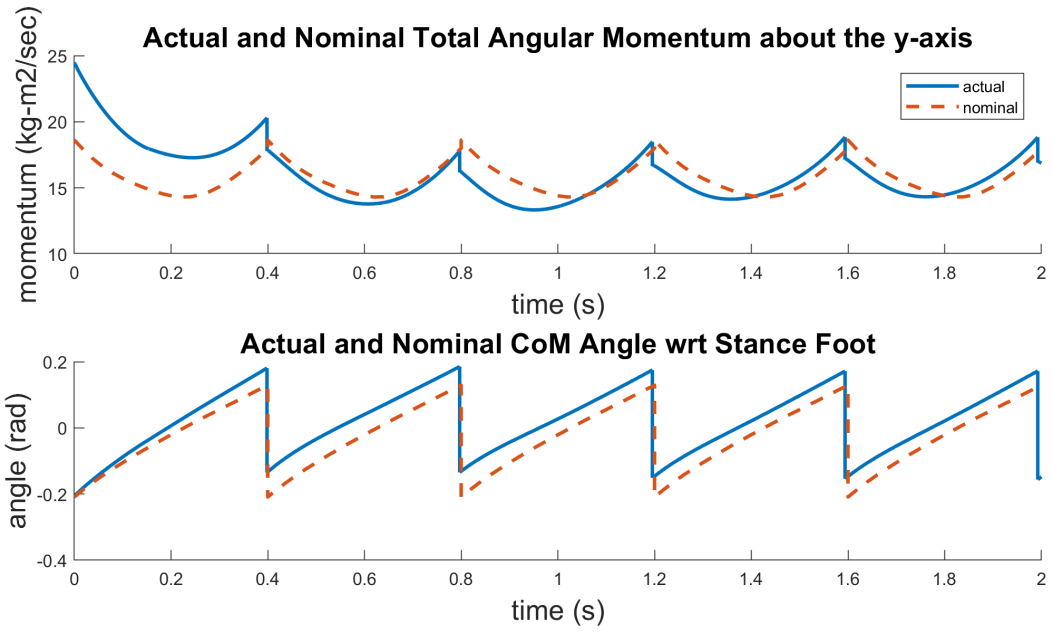


Figure E.6: Nominal and simulated angular momentum and CoM angle over time using MPC to determine stance torque input to stabilize sagittal motion, and the ALIP model to stabilize lateral control with 5.09 kg-m^2/sec of angular momentum perturbation (20% increase) introduced at the start of simulation.

18 cm; however, when the stair height was changed to 22 cm, the simulation converged to a stable orbit. We posit that a more robust implementation of MPC would be able to handle greater variance in stair height than the current simplified implementation, bringing us closer to the goal of having the Cassie biped navigate through a loosely structured cluttered environment.

Appendix F

Biped Bootcamp - Empowering the Next Generation of Roboticists

The Biped Bootcamp was a transformative educational endeavor that originated under the guidance of my esteemed advisor, Professor Jessy Grizzle, and has been an instrumental part of my research journey. The bootcamp is a series fast-paced instructional verbal lessons on the fundalmentals of biped robots that every PhD student that passes through Professor Grizzle's lab undergoes. This personalized and informal learning experience has played a pivotal role in shaping my contributions and achievements in the field of bipedal robotics.

Building upon the foundations laid by Professor Grizzle, I took my extensive written notes from the bootcamp and transformed them into a comprehensive and insightful technical document. This documentation eventually evolved into the Biped Bootcamp technical document—a significant contribution in its own right. This document has served as a valuable resource for learners seeking to understand the fundamentals of controlling and simulating biped robots in Matlab, with its reach extending beyond our lab to inspire and educate others interested in the field. Spanning over 100 pages, the comprehensive document meticulously guides students through various aspects of bipedal robotics, ranging from kinematics and dynamics to control and optimization techniques. The bootcamp's ultimate objective is to equip learners with the necessary knowledge and skills to animate a three-link walker (shown in Figure A.1) using Matlab. The sequel document builds upon the knowledge laid out in the first document and applies it on the more complicated five-link

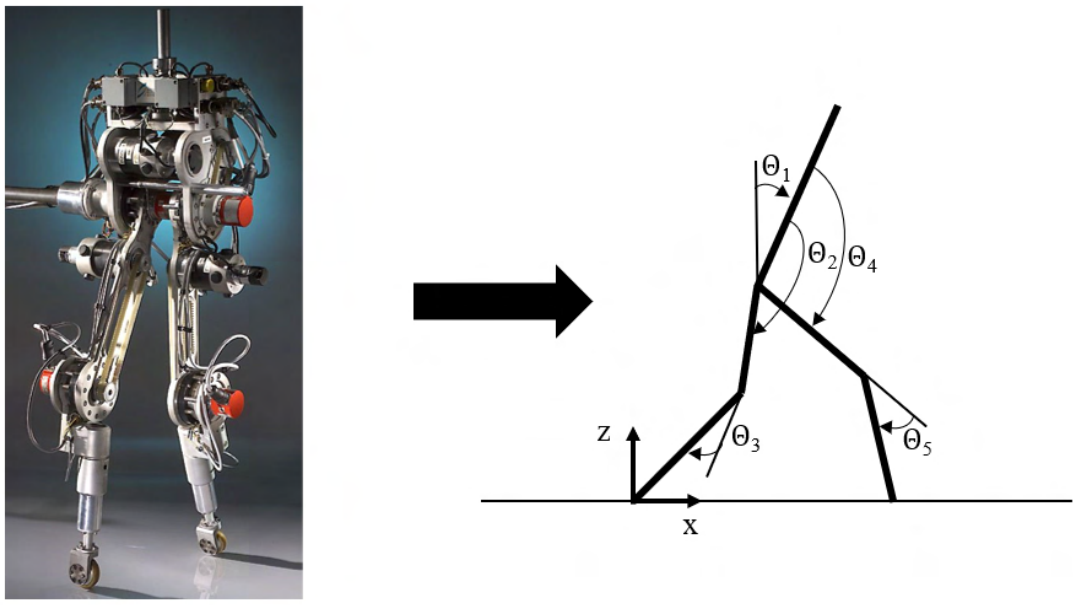


Figure F.1: Actual RABBIT Bipedal Robot (left) and its diagram representation (right).

walker, also known as RABBIT [111] (shown in Figure F.1).

The Biped Bootcamp has had a meaningful impact on the education and career trajectories of numerous students, inspiring a passion for robotics and fostering innovation within the domain. At least fourteen students, encompassing undergraduates, postgraduates, and even a high school student, have benefited from the bootcamp's wealth of knowledge and practical applications.

Ankita Mahajan, one of our talented students, took the learnings from the Biped Bootcamp to a whole new level. Not only did she master the content and apply it to her academic pursuits, but she also went on to share this invaluable knowledge with ten students of her own after adapting the document into Matlab Grader. She has since also further adapted the document into the Julia coding language with the intent to incorporate it into a publicly accessible course. Ankita's adaptation of the bootcamp into the Matlab Grader and Julia programming language highlights

the versatility and flexibility of the material, ensuring it reaches a wider audience and resonates with different student communities.

Another student, Justin Lu, is a testament to the bootcamp's profound impact on nurturing individual creativity and initiative. Armed with the knowledge gained from the bootcamp, Justin embarked on an ambitious project to build his very own quadruped robot. This remarkable endeavor showcases the practical application of the skills acquired during the bootcamp and exemplifies how it has empowered students to venture into new and innovative realms of robotics.

Moreover, the bootcamp's positive impact extends beyond the realm of education and into the professional landscape. Yichen Wang, another talented student, found themselves thriving in a research assistant position in a prosthetics research lab over the summer. The knowledge and skills gained from the bootcamp played a crucial role in preparing Yichen for this research opportunity, highlighting the practical relevance and applicability of the content to real-world challenges.

In conclusion, the Biped Bootcamp stands as a testament to the power of education in nurturing a new generation of roboticists and fostering innovation in the field of bipedal robotics. The dedication, enthusiasm, and accomplishments of our students demonstrate the bootcamp's transformative potential, not only in academic pursuits but also in shaping successful and impactful careers in robotics. The Biped Bootcamp, with its far-reaching effects, has undoubtedly played a pivotal role in cultivating a thriving community of robotic enthusiasts and scholars, paving the way for a future filled with advancements in bipedal robotics and beyond.

Below are the known alumni of the Biped Bootcamp technical document thus far.

Masters Students

Aayushi Shrivastava, M.S., Robotics

Yujie Li, M.S., Robotics

Yufeiyang Gao, M.S., Robotics

Undergraduate Students

Ankita Mahajan, B.S., Electrical Engineering

Gurnoor Kaur, B.S., Robotics, minor in Computer Science

Ishrat Khan, B.S., Robotics

Justin Lu, B.S., Computer Science

Laasya Chukka, B.S., Computer Science

Max Rucker, B.S., Robotics

Miles Bronson, B.S., Computer Science

Victor Popa-Simil, B.S., Biomedical Engineering

Yamato Miura, B.S., Computer Science

Yichen Wang, B.S., Electrical Engineering

High School Students

Vanya Krishna

BIBLIOGRAPHY

[1] Yukai Gong, Ross Hartley, Xingye Da, Ayonga Hereid, Omar Harib, Jiunn-Kai Huang, and Jessy Grizzle. Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway. In *2019 American Control Conference (ACC)*, pages 4559–4566. IEEE, 2019.

[2] Eric R Westervelt, Jessy W Grizzle, Christine Chevallereau, Jun Ho Choi, and Benjamin Morris. *Feedback control of dynamic bipedal robot locomotion*. CRC press, 2018.

[3] DynamicLegLocomotion. Continuous walking on changing incline. https://www.youtube.com/watch?v=ZGtO1RkN1RU, December 2023. Accessed: 2023-12-15.

[4] DynamicLegLocomotion. Cassie walks from stationary flat ground to 20-degree inclined moving treadmill. https://www.youtube.com/watch?v=52nT6H3xkBI, December 2023. Accessed: 2023-12-15.

[5] DynamicLegLocomotion. Cassie on a 0.5 m/s moving walkway. https://www.youtube.com/watch?v=rZZ9UYAjRvc, December 2023. Accessed: 2023-12-15.

[6] DynamicLegLocomotion. Cassie on a 1.2 m/s moving walkway. https://www.youtube.com/watch?v=utANK8jTwuI, December 2023. Accessed: 2023-12-15.

[7] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*, pages 361–389. Springer, 2008.

[8] Tadeusz Mikolajczyk, Emilia Mikołajewska, Hayder F. N. Al-Shuka, Tomasz Malinowski, Adam Kłodowski, Danil Yurievich Pimenov, Tomasz Paczkowski, Fuwen Hu, Khaled Giasin, Dariusz Mikołajewski, and Marek Macko. Recent advances in bipedal walking robots: Review of gait, drive, sensors and control systems. *Sensors*, 22(1212):4440, Jan 2022.

[9] Robert Bogue. Disaster relief, and search and rescue robots: the way forward. *Industrial Robot: the international journal of robotics research and application*, 46(2):181–187, 2019.

[10] Qihao Zhang, Wei Zhao, Shengnan Chu, Lei Wang, Jun Fu, Jiangrong Yang, and Bo Gao. Research progress of nuclear emergency response robot. *IOP Conference Series: Materials Science and Engineering*, 452(4):042102, Dec 2018.

[11] Robert Bogue. Search and rescue and disaster relief robots: has their time finally come? *Industrial Robot: An International Journal*, 43(2):138–143, 2016.

[12] Fauzi Othman, MA Bahrin, N Azli, et al. Industry 4.0: A review on industrial automation and robotic. *J Teknol*, 78(6-13):137–143, 2016.

[13] Ítalo Renan da Costa Barros and Tiago Pereira Nascimento. Robotic mobile fulfillment systems: A survey on recent developments and research opportunities. *Robotics and Autonomous Systems*, 137:103729, Mar 2021.

[14] David Lattanzi and Gregory Miller. Review of robotic infrastructure inspection systems. *Journal of Infrastructure Systems*, 23(3):04017004, Sep 2017.

[15] Rinat Galin and Roman Meshcheryakov. Automation and robotics in the context of industry 4.0: the shift to collaborative robots. In *IOP Conference Series: Materials Science and Engineering*, volume 537, page 032073. IOP Publishing, 2019.

[16] Robert Bogue. Underwater robots: a review of technologies and applications. *Industrial Robot: An International Journal*, 42(3):186–191, Jan 2015.

[17] Maria Kyrarini, Fotios Lygerakis, Akilesh Rajavenkatanarayanan, Christos Sevastopoulos, Harish Ram Nambiappan, Kodur Krishna Chaitanya, Ashwin Ramesh Babu, Joanne Mathew, and Fillia Makedon. A survey of robots in healthcare. *Technologies*, 9(11):8, Mar 2021.

[18] Matthew Spenko, Stephen Buerger, and Karl Iagnemma. *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*. Springer, 2018.

[19] Alexander Stumpf, Stefan Kohlbrecher, David C. Conner, and Oskar von Stryk. Supervised footstep planning for humanoid robots in rough terrain tasks using a black box walking controller. In *2014 IEEE-RAS International Conference on Humanoid Robots*, page 287–294, Nov 2014.

[20] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Tingfan Wu, Daniel Duran, Marshall Floyd, Peter Abeles, Douglas Stephen, Nathan Mertins, Alex Lesman, John Carff, William Rifenburgh, Pushyami Kaveti, Wessel Straatman, Jesper Smith, Maarten Griffioen, Brooke Layton, Tomas de Boer, Twan Koolen, Peter Neuhaus, and Jerry Pratt. Team ihmc's lessons learned from the darpa robotics challenge trials. *Journal of Field Robotics*, 32(2):192–208, 2015.

[21] Duncan Calvert, Bhavyansh Mishra, Stephen McCrory, Sylvain Bertrand, Robert Griffin, and Jerry Pratt. A fast, autonomous, bipedal walking behavior over rapid regions. *arXiv*, Jul 2022. arXiv:2207.08312 [cs].

[22] Robert J. Griffin, Georg Wiedebach, Stephen McCrory, Sylvain Bertrand, Inho Lee, and Jerry Pratt. Footstep planning for autonomous walking over rough terrain. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, page 9–16, Oct 2019.

[23] Lori A Livingston, Joan M Stevenson, and Sandra J Olney. Stairclimbing kinematics on stairs of differing dimensions. *Archives of physical medicine and rehabilitation*, 72(6):398–402, 1991.

[24] Reza Heydari and Mohammad Farrokhi. Model predictive control for biped robots in climbing stairs. In *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*, pages 1209–1214, 2014.

[25] Giorgio Figliolini and Marco Ceccarelli. Climbing stairs with ep-war2 biped robot. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 4, pages 4116–4121. IEEE, 2001.

[26] Giorgio Figliolini, Marco Ceccarelli, and Maurizio Di Gioia. Descending stairs with ep-war3 biped robot. In *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, volume 2, pages 747–752. IEEE, 2003.

[27] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 1405–1411 vol.2, 1991.

[28] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, volume 1, pages 239–246 vol.1, 2001.

[29] Agility Robotics. Cassie robot. https://github.com/agilityrobotics/cassie-doc/wiki.

[30] Jacob Reher, Wen-Loong Ma, and Aaron D. Ames. Dynamic walking with compliance on a cassie bipedal robot. In *2019 18th European Control Conference (ECC)*, pages 2589–2595, 2019.

[31] Yukai Gong and Jessy Grizzle. Zero dynamics, pendulum models, and angular momentum in feedback control of bipedal locomotion, 2021.

[32] Zhaoming Xie, Glen Berseth, Patrick Clary, Jonathan Hurst, and Michiel van de Panne. Feedback control for cassie with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1241–1246, 2018.

[33] Xiaobin Xiong and Aaron Ames. 3-d underactuated bipedal walking via h-lip based gait synthesis and stepping stabilization. *IEEE Transactions on Robotics*, 38(4):2405–2425, 2022.

[34] William Yang and Michael Posa. Impact invariant control with applications to bipedal locomotion. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5151–5158, 2021.

[35] Grant Gibson, Oluwami Dosunmu-Ogunbi, Yukai Gong, and Jessy Grizzle. Terrain-adaptive, alip-based bipedal locomotion controller via model predictive control and virtual constraints. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6724–6731. IEEE, 2022.

[36] Joris Verhagen, Xiaobin Xiong, Aaron Ames, and Ajay Seth. From human walking to bipedal robot locomotion: Reflex inspired compensation on planned and unplanned downsteps, 2022.

[37] Oluwami Dosunmu-Ogunbi, Aayushi Shrivastava, Grant Gibson, and Jessy Grizzle. Stair climbing using the angular momentum linear inverted pendulum model and model predictive control. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6724–6731. IEEE, 2023.

[38] Jonah Siekmann, Kevin Green, John Warila, Alan Fern, and Jonathan W. Hurst. Blind bipedal stair traversal via sim-to-real reinforcement learning. *CoRR*, abs/2105.08328, 2021.

[39] Min Dai, Xiaobin Xiong, and Aaron Ames. Bipedal walking on constrained footholds: Momentum regulation via vertical com control. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10435–10441, 2022.

[40] Alphonsus Adu-Bredu, Grant Gibson, and Jessy W Grizzle. Exploring kinodynamic fabrics for reactive whole-body control of underactuated humanoid robots. *arXiv preprint arXiv:2303.04279*, 2023.

[41] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[42] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2020.

[43] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[44] Roy Featherstone and David Orin. Robot dynamics: equations and algorithms. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 826–834. IEEE, 2000.

[45] Ayonga Hereid and Aaron D. Ames. Frost: Fast robot optimization and simulation toolkit. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, September 2017. IEEE/RSJ.

[46] Ram Vasudevan. Hybrid system identification via switched system optimal control for bipedal robotic walking. In *Robotics Research: The 15th International Symposium ISRR*, pages 635–650. Springer, 2017.

[47] Xingye Da and Jessy Grizzle. Combining trajectory optimization, supervised machine learning, and model structure for mitigating the curse of dimensionality in the control of bipedal robots. *The International Journal of Robotics Research*, 38(9):1063–1097, Aug 2019.

[48] Yukai Gong. *Feedback Control of Highly Dynamic 3D Bipedal Locomotion*. PhD thesis, University of Michigan, 2022.

[49] Xiaobin Xiong and Aaron D. Ames. Orbit characterization, stabilization and composition on 3d underactuated bipedal walking via hybrid passive linear inverted pendulum model. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 4644–4651, Nov 2019.

[50] Theresa Klein. *A Neurorobotic Model of Humanoid Walking*. PhD thesis, University of Arizona, Dec 2011.

[51] I. Poulakakis and J.W. Grizzle. The spring loaded inverted pendulum as the hybrid zero dynamics of an asymmetric hopper. *IEEE Transactions on Automatic Control*, 54(8):1779–1793, Aug 2009.

[52] J. Pratt, P. Dilworth, and G. Pratt. Virtual model control of a bipedal walking robot. In *Proceedings of International Conference on Robotics and Automation*, volume 1, page 193–198 vol.1, Apr 1997.

[53] Alessandro Alla and J Nathan Kutz. Nonlinear model order reduction via dynamic mode decomposition. *SIAM Journal on Scientific Computing*, 39(5):B778–B796, 2017.

[54] Deepak Trivedi, Amir Lotfi, and Christopher D Rahn. Geometrically exact dynamic models for soft robotic manipulators. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1497–1502. IEEE, 2007.

[55] Peter Benner and Heike Faßbender. Model order reduction: Techniques and tools. In *Encyclopedia of Systems and Control*, pages 1227–1234. Springer, 2021.

[56] Manuel Beschi, Enrico Villagrossi, Nicola Pedrocchi, and Lorenzo Molinari Tosatti. A general analytical procedure for robot dynamic model reduction. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4127–4132. IEEE, 2015.

[57] Wisama Khalil, J Kleinfinger, and Maxime Gautier. Reducing the computational burden of the dynamic models of robots. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, pages 525–531. IEEE, 1986.

[58] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

[59] Erica Salvato, Gianfranco Fenu, Eric Medvet, and Felice Andrea Pellegrino. Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning. *IEEE Access*, 9:153171–153187, 2021.

[60] Matthew J Powell and Aaron D Ames. Mechanics-based control of underactuated 3D robotic walking: Dynamic gait generation under torque constraints. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 555–560. IEEE, 2016.

[61] Yukai Gong and Jessy Grizzle. One-step ahead prediction of angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-inspired controller. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2832–2838, 2021.

[62] Benjamin Morris and Jessy W Grizzle. Hybrid invariant manifolds in systems with impulse effects with application to periodic locomotion in bipedal robots. *IEEE Transactions on Automatic Control*, 54(8):1751–1764, 2009.

[63] Chenglong Fu and Ken Chen. Gait synthesis and sensory control of stair climbing for a humanoid robot. *IEEE Transactions on Industrial Electronics*, 55(5):2111–2120, 2008.

[64] Stéphane Caron, Abderrahmane Kheddar, and Olivier Tempier. Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 277–283, 2019.

[65] Ayonga Hereid, Omar Harib, Ross Hartley, Yukai Gong, and Jessy W Grizzle. Rapid trajectory optimization using c-frost with illustration on a cassie-series dynamic walking biped. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4722–4729. IEEE, 2019.

[66] Brent Griffin and Jessy Grizzle. Nonholonomic virtual constraints and gait optimization for robust walking control. *The International Journal of Robotics Research*, 36(8):895–922, Jul 2017.

[67] Ayonga Hereid, Eric A. Cousineau, Christian M. Hubicki, and Aaron D. Ames. 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, page 1447–1454, May 2016.

[68] Koushil Sreenath, Hae-Won Park, Ioannis Poulakakis, and J W Grizzle. A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal

walking on mabel. *The International Journal of Robotics Research*, 30(9):1170–1193, Aug 2011.

[69] Kaveh Akbari Hamed, Jeeseop Kim, and Abhishek Pandala. Quadrupedal locomotion via event-based predictive control and qp-based virtual constraints. *IEEE Robotics and Automation Letters*, 5(3):4463–4470, Jul 2020.

[70] Marc H. Raibert. Hopping in legged systems — modeling and simulation for the two-dimensional one-legged case. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(3):451–463, 1984.

[71] Mark S Redfern and Timothy Schumann. A model of foot placement during gait. *Journal of biomechanics*, 27(11):1339–1346, 1994.

[72] Ali Zamani and Pranav A. Bhounsule. Foot placement and ankle push-off control for the orbital stabilization of bipedal robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4883–4888, 2017.

[73] Young-Dae Hong and Bumjoo Lee. Real-time feasible footstep planning for bipedal robots in three-dimensional environments using particle swarm optimization. *IEEE/ASME Transactions on Mechatronics*, 25(1):429–437, 2020.

[74] Dimitrios Kanoulas, Alexander Stumpf, Vignesh Sushrutha Raghavan, Chengxu Zhou, Alexia Toumpa, Oskar Von Stryk, Darwin G. Caldwell, and Nikos G. Tsagarakis. Footstep planning in rough terrain for bipedal robots using curved contact patches. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4662–4669, 2018.

[75] Steven Crews and Matthew Travers. Energy management through footstep selection for bipedal robots. *IEEE Robotics and Automation Letters*, 5(4):5485–5493, 2020.

[76] Xiaobin Xiong, Jenna Reher, and Aaron D. Ames. Global position control on underactuated bipedal robots: Step-to-step dynamics approximation for step planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2825–2831, 2021.

[77] James Marston Fitch, John Templer, and Paul Corcoran. The dimensions of stairs. *Scientific American*, 231(4):82–91, 1974.

[78] Kenneth Nemire, Daniel A Johnson, and Keith Vidal. The science behind codes and standards for safe walkways: Changes in level, stairways, stair handrails and slip resistance. *Applied ergonomics*, 52:309–316, 2016.

[79] Linda Vesela. Staircase-dimensions of stair steps and their deviations of geometrical accuracy. In *IOP Conference Series: Materials Science and Engineering*, volume 471, page 022012. IOP Publishing, 2019.

[80] John Archea, Belinda Lowenhaupt Collins, and Fred I Stahl. *Guidelines for stair safety*. The Bureau, 1979.

[81] Gerald Farin. Algorithms for rational bézier curves. *Computer-aided design*, 15(2):73–77, 1983.

[82] Hamid Sadeghian, Christian Ott, Gianluca Garofalo, and Gordon Cheng. Passivity-based control of underactuated biped robots within hybrid zero dynamics approach. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4096–4101. IEEE, 2017.

[83] M.W. Spong and F. Bullo. Controlled symmetries and passive walking. *IEEE Transactions on Automatic Control*, 50(7):1025–1031, Jul 2005.

[84] Mark W. Spong, Jonathan K. Holm, and Dongjun Lee. Passivity-based control of bipedal locomotion. *IEEE Robotics & Automation Magazine*, 14(2):30–40, Jun 2007.

[85] Marc D Killpack, Ariel Kapusta, and Charles C Kemp. Model predictive control for fast reaching in clutter. *Autonomous Robots*, 40:537–560, 2016.

[86] Enrica Soria, Fabrizio Schiano, and Dario Floreano. Predictive control of aerial swarms in cluttered environments. *Nature Machine Intelligence*, 3(6):545–554, 2021.

[87] Peter Travis Jardine and Sidney N Givigi. A robust model-predictive guidance system for autonomous vehicles in cluttered environments. *IEEE Systems Journal*, 13(2):2034–2045, 2018.

[88] Mohak Bhardwaj, Balakumar Sundaralingam, Arsalan Mousavian, Nathan D Ratliff, Dieter Fox, Fabio Ramos, and Byron Boots. Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation. In *Conference on Robot Learning*, pages 750–759. PMLR, 2022.

[89] Camille Brasseur, Alexander Sherikov, Cyrille Collette, Dimitar Dimitrov, and Pierre-Brice Wieber. A robust linear mpc approach to online generation of 3d biped walking motion. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 595–601, 2015.

[90] MeiYi Jin. *Trajectory Generation for Biped Robot Climbing Variable Height Stairs Using MPC*. PhD thesis, Hanyang University, 2021.

[91] Jorge Nocedal and Stephen J Wright. Quadratic programming. *Numerical optimization*, pages 448–492, 2006.

[92] BN Biswas, Somnath Chatterjee, SP Mukherjee, and Subhradeep Pal. A discussion on euler method: A review. *Electronic Journal of Mathematical Analysis and Applications*, 1(2):2090–2792, 2013.

[93] Ross Hartley, Maani Ghaffari, Ryan M Eustice, and Jessy W Grizzle. Contact-aided invariant extended kalman filtering for robot state estimation. *The International Journal of Robotics Research*, 39(4):402–430, Mar 2020.

[94] Pengcheng Shi, Zhikai Zhu, Shiying Sun, Xiaoguang Zhao, and Min Tan. Invariant extended kalman filtering for tightly coupled lidar-inertial odometry and mapping. *IEEE/ASME Transactions on Mechatronics*, page 1–12, 2023.

[95] Victor Dhédin, Haolong Li, Shahram Khorshidi, Lukas Mack, Adithya Kumar Chinnakkonda Ravi, Avadesh Meduri, Paarth Shah, Felix Grimminger, Ludovic Righetti, Majid Khadiv, and Joerg Stueckler. Visual-inertial and leg odometry fusion for dynamic locomotion. *arXiv*, Oct 2022. arXiv:2210.02127 [cs].

[96] Jiunn-Kai Huang and Jessy W. Grizzle. Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain. *IEEE Transactions on Robotics*, pages 1–18, 2023.

[97] Jinze Liu, Minzhe Li, Jiunn-Kai Huang, and Jessy W. Grizzle. Realtime safety control for bipedal robots to avoid multiple obstacles via clf-cbf constraints. *arXiv*, Jan 2023. arXiv:2301.01906 [cs].

[98] Jiunn-Kai Huang, Yukai Gong, Dianhao Chen, Jinze Liu, Minzhe Li, Jianyang Tang, Lu Gan, Ray Zhan, Wami Ogunbi, and Jessy Grizzle. Fully Autonomous on the Wave Field 2021. https://youtu.be/gE3Y-2Q3gco, 2021.

[99] Lu Gan. *Semantic-Aware Robotic Mapping in Unknown, Loosely Structured Environments*. Thesis, University of Michigan, 2022. Accepted: 2022-05-25T15:20:18Z.

[100] DynamicLegLocomotion. Cassie walks up stairs in simmechanics. https://www.youtube.com/watch?v=TX74b2i9p-k, December 2023. Accessed: 2023-12-15.

[101] Joel A. E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, Mar 2019.

[102] Joel A.E. Andersson and James B. Rawlings. Sensitivity analysis for nonlinear programming in casadi*. *IFAC-PapersOnLine*, 51(20):331–336, 2018. 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018.

[103] DynamicLegLocomotion. Cassie walking up to 2.2 m/s on flat treadmill. https://www.youtube.com/watch?v=MH8TWk1SjGM, December 2023. Accessed: 2023-12-15.

[104] DynamicLegLocomotion. Cassie walks from stationary flat ground to 4-degree inclined moving treadmill. https://www.youtube.com/watch?v=ZnPoVpUDWQs, December 2023. Accessed: 2023-12-15.

[105] DynamicLegLocomotion. Cassie walks from stationary flat ground to 8-degree inclined moving treadmill. https://www.youtube.com/watch?v=DnYR85WktXk, December 2023. Accessed: 2023-12-15.

[106] DynamicLegLocomotion. Cassie walks from stationary flat ground to 15-degree inclined moving treadmill. https://www.youtube.com/watch?v=0DaqhGlrx38, December 2023. Accessed: 2023-12-15.

[107] Indraswari Kusumaningtyas and Gabriel Lodewijks. Accelerating moving walkway: A review of the characteristics and potential application. *Transportation research part A: policy and practice*, 42(4):591–609, 2008.

[108] DynamicLegLocomotion. Cassie on a 0.8 m/s moving walkway. https://www.youtube.com/watch?v=1MAp6qj8pdM, December 2023. Accessed: 2023-12-15.

[109] Xiao-Bing Kong, Ying-jie Chen, and Xiang-Jie Liu. Nonlinear model predictive control with input-output linearization. In *2012 24th Chinese Control and Decision Conference (CCDC)*, page 688–693, May 2012.

[110] Masoud Soroush and Hossein M. Soroush. Input-output linearizing nonlinear model predictive control. *International Journal of Control*, 68(6):1449–1474, Jan 1997.

[111] Christine Chevallereau, Gabriel Abba, Yannick Aoustin, Franck Plestan, Eric Westervelt, Carlos Canudas De Wit, and Jessy Grizzle. Rabbit: A testbed for advanced control theory. *IEEE Control Systems Magazine*, 23(5):57–79, 2003.