REAL-TIME PERCEPTION AND MIXED-INTEGER FOOTSTEP CONTROL FOR UNDERACTUATED BIPEDAL WALKING ON ROUGH TERRAIN

Brian Acosta

A DISSERTATION

in

Mechanical Engineering and Applied Mechanics

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2025

Supervisor of Dissertation

Michael Posa, Assistant Professor of Mechanical Engineering and Applied Mechanics

Graduate Group Chairperson

Prashant K. Purohit, Professor of Mechanical Engineering and Applied Mechanics

Dissertation Committee

Dan Koditschek, Alfred Fitler Moore Professor of Electrical and Systems Engineering
Michael Posa, Assistant Professor of Mechanical Engineering and Applied Mechanics
Antonio Locquercio, Assistant Professor of Electrical and Systems Engineering
Xiaobin Xiong, Assistant Professor of Mechanical Engineering, University of Wisconsin

REAL-TIME PERCEPTION AND MIXED-INTEGER FOOTSTEP CONTROL FOR UNDERACTUATED BIPEDAL WALKING ON ROUGH TERRAIN

COPYRIGHT

2025

Brian Jeffrey Acosta

ACKNOWLEDGEMENT

I want to first thank my advisor, Michael Posa. Michael gave me broad latitude to explore research directions I was interested in, and pushed me to clearly articulate the problems I was solving, why they were hard, and why they mattered. Michael is deeply kind without mincing words on technical matters, and I hope to replicate his patience and high standards as I continue in my career.

I also with to thank my fellow DAIR Lab members. Mathew Halm provided me valuable mentorship and ecouragement early in my PhD, and his impeccable food and coffee recommendations made Philadelphia feel like home. Yu-Ming Chen wrote much of the original Cassie control stack from scratch, and taught me much of what I know about writing (and debugging) robot controllers. My frequent discussions with Will Yang on how to implement things "the right way" and his high standards for controller robustness greatly improved my software engineering abilities and the perceptive locomotion results in this thesis. Alp Aydinoglu gave me helpful advice on mixed-integer programming, and, along with Leon Kim, Bibit Bianchini, Hien Bui, and Grey Sarmiento, could be relied on for insightful questions and hot takes. Without the safety-tether carrying talents of the above mentioned lab-mates, as well as Chandravan Kunjeti, Wei-Cheng Huang, and Minku Kim, none of the Cassie experiments in this thesis would have been possible.

Special thanks to everyone who helped keep Cassie working. To Peter Szczesniak, Jason Pastor, and Ari Bortman in the MEAM machine shop for always being enthusiastic to make a part or lend a tool. Thank you to Nathan Peterman and Oluwami Dosunmu-Ogunbi for helping me understand Cassie's battery requirements.

Thank you to my committee, Professors Dan Koditschek, Xiaobin Xiong, and Antonio Loquercio, for your insightful feedback.

I would be remiss not to also thank the Drake team at Toyota Research Institute. Every slack conversation with Jeremy Nimmer taught me something new about writing software, and Drake provided an invaluable scaffolding upon which to implement the algorithms developed in this thesis.

Purdue Mechanical Engineering provided me the foundational skills and community I needed to launch a research career in robotics. Professor Song Zhang was my first controls professor and research advisor. He taught me how to write effectively and how to contextualize a research problem in a broader field. Without the advice and intellectual partnership of Naveed Riaziat, I would not have applied to nor been accepted to Penn.

I would like to thank my parents and brothers for their support of my long career as a student – especially my late father, who encouraged me to be broadly and endlessly curious.

Thank you to Sydney, for everything. Being yours is my greatest accomplishment.

ABSTRACT

REAL-TIME PERCEPTION AND MIXED-INTEGER FOOTSTEP CONTROL FOR UNDERACTUATED BIPEDAL WALKING ON ROUGH TERRAIN

Brian Acosta

Michael Posa

The promise of bipedal robots is to go where people go, serving as surrogates for human labor in dangerous unstructured environments. For the most part, this promise remains unrealized, due partially to the difficulty of controlling bipedal locomotion in these environments. The primary challenge for controlling bipedal locomotion is underactuation. Standing on a single leg limits control authority, requiring appropriate foot placement to generate or absorb momentum and maintain balance. Rough terrain exacerbates this challenge by introducing restrictions on where the robot can step. These restrictions must be identified from onboard sensing modalities and accounted for in the footstep plan, all while meeting the strict real-time requirements of feedback control. In this thesis, we examine systems, modeling choices, and algorithms for solving this problem, ultimately enabling dynamic bipedal walking over previously unseen discontinuous terrain.

Conventional approaches decouple the problem of walking over rough terrain into separate modules for footstep planning and motion control, limiting walking speed and online adaptability. The beginning of this thesis introduces a new model-predictive-control-style footstep planner which eliminates this decomposition. We jointly optimize over the robot's dynamics and discrete choice of stepping surface in real time to stabilize underactuated walking over constrained footholds.

Our footstep controller depends on approximating the safe terrain as a union of convex planar polygon "stepping stones". In order to generate such an approximation from onboard sensors in real time, we propose novel safe terrain segmentation and convex decomposition algorithms. Our segmentation approach avoids the common design choice of plane segmentation, which we argue makes segmentation algorithms slower and less reliable. Instead, we classify terrain as safe based only on

local features, yielding a segmentation which is both fast to compute and temporally consistent. We present full stack perceptive locomotion experiments on the underactuated biped Cassie, leveraging our novel footstep controller and perception pipeline to walk over previously unseen discontinuous terrain.

Finally, we present an exploratory study of a cascaded-fidelity model predictive footstep controller, which combines elements of our first footstep planner with whole-body model predictive control in order to navigate even more challenging terrains.

TABLE OF CONTENTS

ACKNOWLEDGEMENT iii

ABSTRACT v

LIST OF TABLES x

LIST OF ILLUSTRATIONS xi

CHAPTER 1 : Introduction 1

1.1 Contributions and Outline 3

CHAPTER 2 : Bipedal Robot Terminology 5

CHAPTER 3 : Related Work 7

3.1 Controlling Dynamic Bipedal Locomotion 7

3.2 Footstep Planning on Discontinuous Terrain 11

3.3 Identifying Safe Terrain from Perception 13

CHAPTER 4 : Background 15

4.1 Cassie 15

4.2 Hierarchical Control of Bipedal Walking 18

4.3 Operational Space Control 26

4.4 Model Predictive Control 27

4.5 Robot-Centric Elevation Mapping 30

CHAPTER 5 : Model Predictive Footstep Control 31

5.1 MPFC Problem Statement 32

5.2 Output tracking via Operational Space Control 37

5.3 Results 41

5.4 Conclusion 49

CHAPTER 6 : Stable Steppability Segmentation and Convex Decomposition 51
6.1 Stable Steppability Segmentation 54
6.2 Convex Planar Decomposition 57
6.3 Results 60
6.4 Discussion 67
6.5 Conclusion 68

CHAPTER 7 : Full-Stack Perceptive Locomotion Experiments 70
7.1 Experimental Setup 70
7.2 Results 74
7.3 Discussion 80
7.4 Conclusion 82

CHAPTER 8 : Cascaded Fidelity MPFC 83
8.1 The Alternating Direction Method of Multipliers 84
8.2 ALIP MPFC ADMM Formulation 86
8.3 Cascaded-Fidelity MPFC formulation 90
8.4 Evaluation 100
8.5 Discussion 105
8.6 Conclusion 106

CHAPTER 9 : Conclusions and Future Work 107
9.1 Lessons Learned 108
9.2 Future Work 109

APPENDIX A : ALIP MPFC IMPLEMENTATION DETAILS 111

APPENDIX B : MPFC AND S3 PARAMETERS 113

APPENDIX C : ADMM DERIVATION WITH WEIGHTED PROJECTION 115

APPENDIX D : CASCADED FIDELITY MPFC GAINS 117

BIBLIOGRAPHY 119

LIST OF TABLES

TABLE 5.1 Stepping Stone Parameter Distributions 45
TABLE 5.2 MPFC Solve-Time Statistics (134,654 Solves) 46

TABLE 6.1 Comparison of S3 and Plane Segmentation Baselines As Benchmarked 63
TABLE 6.2 Moving Obstacles Segmented by S3 vs. Hysteresis parameter ↑. An obstacle is defined as segmented out if it creates a hole in the terrain segmentation before it comes to rest. 67

TABLE 8.1 Whole-Body MPC Timing Parameters 93
TABLE 8.2 Computation Time Breakdown for CF-MPFC Iterations 105

TABLE B.1 MPFC Parameters 113
TABLE B.2 OSC Gains (Hardware) 114
TABLE B.3 OSC Gains (Simulation) 114
TABLE B.4 Perception Stack Parameters 114

TABLE D.1 CF-MPFC Parameters 117
TABLE D.2 CF-MPFC Whole-Body Cost Weights 118
TABLE D.3 CF-MPFC ALIP cost weights 118

LIST OF ILLUSTRATIONS

FIGURE 4.1 Cassie's configuration space. 15
FIGURE 4.2 The ALIP model assumes that the robot's CoM is restricted to a virtual plane above the terrain. The states of the ALIP model are the horizontal CoM positions, and the angular momentum of the robot about the horizontal axes. . 19
FIGURE 4.3 The ALIP model with mass m, constant CoM height H, and ankle torque in the sagittal plane, u. 20

FIGURE 5.1 Key MPFC decision variables and constraints for a horizon of 2 stance phases. x_c is the current ALIP state, u is ankle torque applied during the current stance phase, x_0 is the ALIP state at the end of the current stance phase, and x_1 is the ALIP state at the end of the next stance phase. The current stance foot position, p_0, is unconstrained, and subsequent footsteps are constrained to lie in either $\mathcal{P}_j$ or $\mathcal{P}_q$ using integer variables. 34
FIGURE 5.2 Relationship between the nominal stance phases and MPFC gait timing optimization. The initial stance duration is adjusted continuously by optimizing over the remaining stance time, T. 34
FIGURE 5.3 To enforce the planarity assumption of the ALIP, we use OSC to drive Cassie's CoM to a virtual plane defined by current and upcoming stance foot positions. 38
FIGURE 5.4 Trajectory from the swing foot position at the beginning of the swing phase, $p_{sw,0}$ to the next footstep solution from MPFC, $p_{sw,des}$. We adapt the direction and clearance of the trajectory's midpoint, p_{mid}, based on the relative positions of $p_{sw,0}$ and $p_{sw,des}$ to ensure sufficient ground clearance. 40
FIGURE 5.5 Cassie walks across an 8 m $\times$ 23 cm beam using MPFC. 42
FIGURE 5.6 Cassie walks up a set of stairs with a 27 cm depth and a 15 cm rise using MPFC. 42
FIGURE 5.7 Velocity tracking for the beam and stairs results shown in Fig. 5.5 and Fig. 5.6 42
FIGURE 5.8 Freeform walking over constrained footholds using MPFC. **Top:** Cassie's pose throughout the freeform walking trial. **Bottom:** Velocity Tracking during the freeform walking trial. 43
FIGURE 5.9 Comparing the velocity tracking performance of MPFC over 40 cm $\times$ 15 cm stairs using a naive cost and the subspace cost we develop in Section 5.1.1. The naive cost (Gait Cost) is on deviation from a reference trajectory, while our subspace cost (Subspace Cost) is on deviation from the subspace of Period-2 ALIP trajectories with the desired velocity. 44
FIGURE 5.10 Example of successfully traversing a random stepping stone environment in simulation with $d_{min} = 35$ cm. 45
FIGURE 5.11 Success rates for walking across randomly-generated stepping stones in simulation. Results with step-timing optimization are labeled Opt-T, and results without step timing optimization are labeled Fixed-T. Step-timing optimization increases success rates over small footholds. 46
FIGURE 5.12 Velocity tracking, solve times, and elevation change for Cassie walking on discontinuous terrain including steps, a curb, and a grassy hill. 47

FIGURE 5.13 Motion tiles showing Cassie ascending and descending steps, stepping over a curb onto the grass, and walking up a grassy slope in one continuous walking trial using our proposed locomotion stack and the perception pipeline proposed in Chapter 6. 48

FIGURE 6.1 Pipeline for converting an elevation map of the terrain into a set of convex polygons which can be used to plan safe footsteps. Our Stable Steppability Segmentation produces a temporally-consistent steppability mask representing a 2D overhead view of the safe terrain. We then extract the boundaries of the safe terrain as non-convex polygons, which we decompose into convex polygons using an algorithm based on approximate convex decomposition. 53

FIGURE 6.2 Block diagram of S3, our proposed terrain segmentation approach. Safety criteria are combined into an overall safety score for each elevation map pixel, before applying hysteresis to enhance temporal consistency. 55

FIGURE 6.3 Walking over ledges with Cassie requires asymmetric constraints on the footstep position, p, which is mapped to the center of Cassie's foot. The desire to specifically avoid stepping below an edge motivated our curvature based safety criterion, which differentiates between sides of an edge using the sign of the elevation map's Laplacian. 56

FIGURE 6.4 The environments used to collect data for benchmarking the perception stack's performance. From top to bottom the terrains are **Lab**, **Brick Steps**, and **Grass** 61

FIGURE 6.5 Offline benchmark of S3 compared to plane segmentation baselines. **Top:** Histogram of the run time of each segmentation algorithm over 60 seconds of elevation mapping data from each test environment. Benchmark was run on an Apple Macbook Pro with an M1 Max CPU, 10 cores, and 64 GB of RAM. S3 has the fastest and most consistent run times. EM_cupy is the slowest, with a highly variable run time, due to the repeated use of RANSAC to refine the plane segmentation. **Bottom:** Histogram of the frame-to-frame IoU of the safe terrain segmentation over the same datasets. Our method reliably achieves a frame-to-frame IoU close to 1 across environments, representing excellent temporal consistency. 64

FIGURE 6.6 Tiles showing the output of each segmentation method for each evaluation environment at 1 second intervals. In the **Lab** and **Grass** environments, the use of Navier-Stokes based inpainting allows S3 to correctly identify the entire elevation map as steppable. The S3 segmentation experiences minimal "flickering" of the steppable terrain compared to the baselines. Animations of these segmentation results can be seen in the supplemental video. 65

FIGURE 6.7 The final convex decomposition has a similarly shaped IoU distribution to the safe terrain segmentation, though is less consistent overall. 66

FIGURE 6.8 Frame-to-Frame IoU of the S3 terrain segmentation results with varying levels of hysteresis, evaluated on the **Brick Steps** data. The lowest observed IoU was 0.78, even without hysteresis, in contrast to both of the plane segmentation baselines, whose the IoU varied across the entire [0, 1] interval. 68

FIGURE 7.1 The perception and control stack proposed in this chapter to achieve underactuated walking over discontinuous terrain. Our perception stack (A) generates convex polygon foothold constraints for MPFC, a mixed-integer MPC style footstep planner (B). MPFC sends the next footstep, step timing adaptation, and ankle torque plan to a low-level operational-space-control process (C) which performs kHz level torque control. 72
FIGURE 7.2 The experiment setup for the Physical Cassie robot showing which computers run which processes. The robot is attached to a safety tether which remains slack while the robot is walking. 73
FIGURE 7.3 Success rates for walking across randomly-generated stepping stones in simulation, where the stepping stone sizes are distributed according to Table 5.1 based on the minimum side lenght, d_{min}. Results with step-timing optimization are labeled Opt-T, and results without step timing optimization are labeled Fixed-T. We report results with ground truth state and terrain (GT), as well as with perceptive terrain using S3 (Perceptive). The lower success rate using perception is primarily due to isotropic safety margin in S3 reducing the lateral steppable area compared to ground-truth, which accounts for the width and length of the foot separately. 75
FIGURE 7.4 MPFC simulation experiments using ground truth vs. perceptive terrain. **Top:** we use ground truth terrain information to walk over a 23 cm wide beam, and stairs with a rise of 15 cm and a depth of 27 cm. **Bottom:** Displaying the elevation map for walking over the same terrain types using S3. Safety margin in S3 results in less steppable area than for ground truth, so the minimum traversable dimensions are increased to 35 cm wide for the beam and 40 cm deep for the stairs. 76
FIGURE 7.5 Cassie walks up and down brick steps using the perception and control framework developed in this thesis. Left: the physical robot and steps. Middle: an elevation map of the steps. Right: the convex decomposition of the safe terrain. 77
FIGURE 7.6 Another view of Cassie descending a set of steps using our full-stack perceptive locomotion architecture. 78
FIGURE 7.7 The top of the brick steps has the largest height change at 16 cm, which sometimes challenges the OSC's ability to track the CoM and swing foot, and creates large impacts when stepping down. 79
FIGURE 7.8 Cassie steps over a high curb onto a grassy knoll. 79
FIGURE 7.9 Our proposed system naturally handles terrain with no obvious planar approximation. Despite constantly varying height and surface normals, S3 classifies the entirety of this sinusoidal terrain as safe, resulting in a single steppable region matching the extents of the map. Because ALIP dynamics are height-independent, the z-coordinate of each planned footstep does not affect the rest of the MPC solution. Therefore, we use the elevation map to determine the footstep height sent to the low-level OSC. 81

FIGURE 8.1 We grid search over the ADMM parameters which give the best closed-loop reliability for walking over stepping stones with ALIP MPFC. We ultimately choose $m = 0, \rho = 0.1$. 88

FIGURE 8.2 Success rate vs. planning horizon for traversing random stepping stones in simulation. We perform 100 trials, randomizing the stepping stone dimensions and positions according to the distributions in Table 5.1. 89
FIGURE 8.3 Distributions of solve times for ADMM vs. MIQP implementations of ALIP MPFC walking over random 35-40 cm stepping stones. We compile the solve times over 100 trials for each combination of solution method and planning horizon, resulting in over 100,000 data points per box plot - the exact number varies due to some simulations terminating early due to a fall. Solve times above the 99.9^{th} percentile are shown as outliers. Simulations were performed on a MacBook Pro with an Apple M1 Max CPU and 64 GB of RAM. The MIQP was solved with Gurobi using 4 threads, while ADMM was solved in a single thread, using OSQP for the ADMM iteration sub-problems. 90
FIGURE 8.4 Model schedule, footstep schedule, and gait schedule for Cascaded-Fidelity MPFC. 96
FIGURE 8.5 Comparison of the control architectures for ALIP MPFC (top) and CF-MPFC (bottom). 101
FIGURE 8.6 ALIP MPFC fails to complete a 20 cm step up due to the swing foot not making it on to the step. Relying on a heuristic swing foot planner places a high burden on the OSC tracking performance compared to CF-MPFC, which accounts for the robot's dynamics when planning a swing foot motion. 102
FIGURE 8.7 Cassie successfully traversing up to 30 cm height changes using CF-MPFC in simulation. 103
FIGURE 8.8 Cassie traverses a narrow beam, stepping stones, and stairs using CF-MPFC. . 104

CHAPTER 1

Introduction

Bipedal robots can theoretically traverse challenging terrain by making and breaking contact with the ground to step over obstacles. This capability suggests their fitness for applications like disaster response, planetary exploration, and deployment in cluttered home environments. Despite these morphological capabilities, walking over real-world rough terrain remains a key challenge in controlling bipedal robots (Gu et al., 2025). From a planning and control perspective, rough terrain is challenging because foot placement is key to stabilizing the underactuated dynamics of bipedal locomotion (Pratt et al., 2006; Raibert and Brown, 1983), yet unsafe terrain creates non-trivial, non-convex restrictions on where the robot can step. From a sensing perspective, the robot must identify safe terrain only through onboard sensing modalities such as lidar or RGB-D cameras. Finally, robots must continually update their control strategy and terrain estimation to account for disturbances, model error, changing terrain, and the limited range of onboard range sensors. This places hard real time requirements on control and perception algorithms for walking on rough terrain. In response to these challenges, we present advances in robot perception and control, resulting in a real-time perceptive locomotion framework which expands the capabilities of bipedal robots to dynamic walking over previously unseen discontinuous terrain.

Focusing first on planning and control, walking over non-convex, potentially disconnected terrain has traditionally been addressed by decomposing the problem into a footstep planner and a locomotion controller (Chestnutt et al., 2003; Kuindersma et al., 2016). This lets the non-convex aspects be handled in a low-dimensional space through classic motion planning algorithms like graph search (Kuffner et al., 2003; Griffin et al., 2019), but results in slow walking speeds to avoid falling, since the dynamics of the robot are not incorporated into the footstep planner. Other approaches allow robot motion and footstep positions to be continuously re-planned subject to a pre-specified sequence of convex "stepping stone" constraints on the foot placement (Nguyen et al., 2016; Dai et al., 2022), requiring an additional high-level contact planner or requiring the stepping stones to have a well-

defined order. Pre-specifying the foothold sequence eliminates the ability to adapt the foothold choice online in response to disturbances, and risks sub-optimality or infeasibility of the lower level control problem. In this thesis, we introduce a real time model-predictive footstep controller which jointly plans over the discrete stepping stone choice and continuous footstep positions, subject to the dynamics of an underactuated walking model. Our controller is a complete solution which plans and stabilizes free-form walking over constrained footholds, requiring only a list of stepping stones and a velocity command as input.

To apply our controller on hardware, we require a method for generating the stepping stone constraints online. Existing approaches extract planar features from either depth maps (Mishra et al., 2021), point clouds (Bertrand et al., 2020), or elevation maps (Miki et al., 2022b) on a frame-by frame basis. This approach is vulnerable to even mild non-planarity of the terrain, and the difficulty of incorporating history into the segmentation results in stepping stones which flicker into and out of existence (Corbères et al., 2023; Calvert et al., 2022). Even when the real terrain is actually planar, impacts and state estimate drift, both common in bipedal walking, can introduce artifacts which make the terrain appear non-planar, making these methods unusable (Grandia et al., 2022). In this thesis, we take the perspective that planar stepping stones are an approximation of the safe terrain which is convenient for control, but that strictly enforcing planarity makes segmentation algorithms brittle in practice. In response to this brittleness, we develop a novel terrain segmentation approach for elevation maps, considering only criteria which are local to the neighborhood of each elevation mapping cell. This structural change eliminates the issue of flickering stepping stones, and makes it easy to incorporate history into the segmentation.

Our initial controller formulation for walking over stepping stones uses the linear inverted pendulum as a model of underactuated walking (Kajita et al., 2001). This requires the assumptions of planar terrain and negligible centroidal angular momentum. Large step-to-step height changes prohibitively violate these assumptions by requiring large accelerations of the center of mass and swing foot. In order to traverse terrains with large height changes, we propose a cascaded-fidelity MPC formulation with stepping stone constraints. This formulation leverages whole-body dynamics se-

quentially composed with linear-inverted-pendulum dynamics to generate more dynamic behaviors while efficiently reasoning about the long-term consequences of it's footstep choices. In developing the cascaded-formulation, we show that local solvers are effective for real-time locomotion control over constrained footholds with longer planning horizons than mixed-integer approaches.

1.1. Contributions and Outline

We start by reviewing some useful bipedal robot terminology in Chapter 2. In Chapter 3 we discuss related work, including footstep planning, controlling dynamic bipedal walking, and algorithms and representations for safe terrain identification. Chapter 4 contains the necessary technical background for the following chapters.

The core contributions of this thesis begin in Chapter 5. We first present Model Predictive Footstep Control (MPFC), a new walking controller which allows underactuated bipeds to perform free-form walking over constrained footholds (Acosta and Posa, 2023). This controller assumes a linear reduced order model of the robot's dynamics and a convex polygon terrain representation to formulate footstep planning as a mixed integer quadratic program. The resulting optimization problem can be solved in a receding horizon fashion in real time, and generates a fully-coupled plan over discrete choice of foothold, footstep positions, reduced-order dynamics, gait timing adaptation, and ankle torque. We demonstrate MPFC traversing discontinuous terrains in simulation and on hardware using the bipedal robot Cassie.

In Chapter 6, we propose a new approach for terrain segmentation and convex polygon decomposition. We establish a general framework, which we call "Stable Steppability Segmentation" for classifying the safety of each cell of an elevation map via a composition of safety criteria. We instantiate specific safety criteria based on experimental observations. We validate our approach to be both more computationally efficient and temporally consistent than explicit plane segmentation using real-world perceptive locomotion data from several terrain types. We propose an approach based on approximate convex decomposition (Lien and Amato, 2006) to convert the resulting steppability mask into a set of convex planar polygons approximating the safe terrain around the robot.

Chapter 7 presents full-stack perceptive locomotion experiments on the bipedal robot Cassie, incorporating the contributions of the previous two chapters into the first hardware demonstration of perceptive model-based control of Cassie over discontinuous terrain.

Finally, Chapter 8 increases the robustness and generality of MPFC. First, we propose a locally optimal solver which can handle longer planning horizons in real time than the MIQP formulation from Chapter 5, increasing the closed-loop robustness of the controller. Then, we use this solver within a sequential quadratic programming framework to solve a cascaded-fidelity MPFC formulation. By considering the robot's full dynamics, the cascaded-fidelity MPFC is capable of traversing larger height changes that the original MPFC.

We conclude by discussing directions for future research.

CHAPTER 2

Bipedal Robot Terminology

We provide informal definitions of some biped-specific jargon which will be useful to review before discussing prior works, deferring the full technical background to Chapter 4.

A bipedal robot has two legs, each of which terminates at a *foot*. A foot can be its own link, or it can be a *point foot*, a point at the end of a structural link which makes contact with the ground. In order to walk, a biped makes and breaks contact with the ground, alternating between phases of *single support* (a.k.a. *single stance*), when one foot is in contact with the ground, and *double support* (a.k.a. *double stance*), when both feet are in contact with the ground. The double-support phases can have a duration of zero, in which case the robot alternates directly between *left stance* and *right stance*. Gaits with different contact sequences, such as running, skipping, or jumping, will not be addressed in this thesis. During single support phases, the foot in contact with the ground is called the *stance foot*, and the foot which is suspended in the air by the opposite leg is called the *swing foot*. The transition from single stance to double stance is sometimes called *touchdown*, and the transition from double to single stance is called *liftoff*.

The weight of the robot is supported by the vertical component of the *ground reaction forces*, which act on the robot's feet and, together with gravity, determine how the linear and angular momenta of the robot change with time. The point where the net ground reaction force acts on the foot is the *center of pressure (CoP)*, which, for flat ground, is equivalent to the *Zero-Moment-Point (ZMP)*, so named because placing the net ground reaction force at the ZMP results in zero net moments about the horizontal axes of the foot (Goswami, 1999). The location of the CoP is bounded by the *support polygon*, the convex hull of all of the points which are in contact with the ground at a given time.

This thesis concerns the control of *dynamic* or *underactuated* walking. We define dynamic walking similarly to Westervelt et al. (2007) as a walking motion where the CoP is often or always on the

boundary of the support polygon. This could be because the robot is using a heel-toe walking pattern, or, in our case, because the robot has feet with no area, such as point feet or blade feet. A corollary of this definition is that at least one of the biped's degrees of freedom is both unactuated and unstable. Stabilizing such degree(s) of freedom is the core challenge of dynamic walking, and is most often accomplished through *footstep control*, where the location of the foot relative to the robot's CoM at touchdown is used to regulate the underactuated degrees of freedom.

This thesis expands the capabilities of underactuated bipeds by providing an algorithm for footstep control when there are non-convex constraints on the *footstep location*. We distinguish a *footstep location* (or simply a *footstep*), which is the actual or planned 3-Dimensional position of the robot's stance foot, from a *foothold*, which is a region on the ground where it is valid for the robot to step.

CHAPTER 3

Related Work

This chapter reviews related work for perceptive, dynamic, bipedal walking on rough terrain. First, we review prior methods for controlling dynamic bipedal and legged locomotion, including reduced order models, trajectory optimization, whole-body model predictive control, reinforcement learning, and methods which specifically handle rough terrain. Then, we review methods of footstep planning over rough terrain, and methods for identifying safe terrain from perception.

3.1. Controlling Dynamic Bipedal Locomotion

We start by reviewing the control methods which have been proposed to stabilize dynamic bipedal locomotion, or in many cases, general legged robot locomotion. While we cover a wide spectrum of methods, we note that the works referenced in this section make the simplifying assumption of a fixed gait schedule. That is, the order in which the robot's feet make and break contact with the ground is set to a predetermined sequence. Real time contact-implicit control is an active area of research with several interesting examples in legged locomotion (Kim et al., 2024) and robot manipulation (Aydinoglu et al., 2023), but has not been as widely studied as the types of methods we cover below.

3.1.1. Reduced Order Models

Simplified dynamics models are often used to manage the computational complexity of controlling legged robots. These models relate footstep locations, center-of-mass dynamics, and ground reaction forces, while ignoring the robot's internal degrees of freedom. Raibert's experiments on controlling monopedal hoppers (Raibert et al., 1984) inspired the development of the Spring Loaded Inverted Pendulum (SLIP) (Blickhan, 1989) as biomechanical model for dynamic running and walking behaviors (Full and Koditschek, 1999; Geyer et al., 2006). SLIP was subsequently used as a control model for dynamic legged locomotion for other simple hoppers (Hutter et al., 2010) as well as complex high-degree of freedom bipeds (Wensing and Orin, 2013b; Piovan and Byl, 2016; Apgar et al., 2018; Yang and Posa, 2023). Perhaps the most widely used reduced order model is the Linear In-

verted Pendulum (LIP) (Kajita et al., 2001), which assumes the robot is a point mass at a fixed height, yielding linear dynamics.

Combined with some form of output tracking, the LIP model and its variants have formed the basis of numerous dynamic walking controllers. Xiong and Ames (2022) developed a *step to step* LIP model, which integrates the LIP dynamics over each stance phase to transform dynamic walking via footstep control into a discrete-time LTI control synthesis problem. Powell and Ames (2016) introduced angular momentum about the contact point as a LIP velocity coordinate due to its invariance to rigid body impacts. Gong and Grizzle (2021) showed that this coordinate choice, which they named the ALIP model, improves the predictive accuracy of the LIP model during single stance, even for robots with significant unmodeled leg mass. ALIP based control strategies have stabilized underatuated walking over predefined footstep locations (Dai et al., 2022), slopes (Gibson et al., 2022), and simulated staircases (Dosunmu-Ogunbi et al., 2023). We derive several solutions to the ALIP dynamics in Chapter 4, and use them for control synthesis in Chapter 5.

3.1.2. Offline Trajectory Optimization

Because reduced order models inherently limit the motions which can be realized, offline trajectory optimization (Hereid et al., 2016; Posa et al., 2016) has been used to generate libraries of reference motions. Reher et al. and Gong et al. exploit the hybrid zero dynamics (HZD) framework (Westervelt et al., 2003) to design gait libraries for velocity tracking on the Cassie biped (Reher and Ames, 2021; Gong et al., 2018). HZD provides stability guarantees for planar bipedal walking, but in practice requires additional footstep correction to achieve stable walking in 3D (Reher et al., 2016). High-accuracy trajectory optimization is too slow to be deployed online, rendering it unsuitable for walking on unstructured terrain, where obstacles will necessitate generating new motions on the fly. for walking on flat ground with perturbation recovery, Gu et al. developed a library of 200 reference gaits, including cross-legged recovery maneuvers (Gu et al., 2022).

3.1.3. Whole-Body MPC

Recent years have seen algorithms capable of optimizing over a full-order Lagrangian model of the robot dynamics in real time (Mastalli et al., 2023). To solve these the nonlinear trajectory optimization problems quickly online, these works use solvers that create sequential linear-quadratic approximations of a nonlinear trajectory optimization problem, such as iterative LQR (Li and Todorov, 2004) or sequential quadratic programming (Boggs and Tolle, 1995). Most formulations enforce inequality constraints with penalty methods, to enable fast solve times by leveraging the Ricatti structure of optimal control (Grandia et al., 2022; Howell et al., 2019). However, recent efforts by Khazoom et al. (2024) have explored the use of general purpose ADMM-Based QP solvers for the SQP sub-problems to incorporate general inequality constraints. In order to more accurately approximate the value function of an infinite-horizon optimal control problem, despite the short horizons necessary for real-time whole-body MPC, Li and Wensing propose appending a reduced-order model MPC problem to the end of the whole-body MPC problem (Li and Wensing, 2024). They also propose to improve the low level tracking performance by incorporating the MPC value function into the cost function of their whole-body-control QP. In the context of legged robots, real-time implementation of whole-body MPC relies on dedicated software for efficient evaluation of dynamics algorithms and their partial derivatives, e.g. (Carpentier et al., 2019; Todorov et al., 2012).

3.1.4. Reinforcement Learning

Reinforcement learning (RL) is an optimal control method where a control policy is optimized through repeated interaction with the environment. During these interactions, the policy accrues a numerical reward signal by applying beneficial actions, and the optimizer attempts to maximize the expected reward achieved by executing the policy (Sutton and Barto, 2018). RL has been applied to bipedal locomotion since the work of Benbrahim and Franklin (1997), who used RL to learn a central pattern generator and auxiliary corrective controllers. The success of RL for controlling legged locomotion hit an inflection point with the introduction of domain randomization (Tobin et al., 2017), which lessened the impact of the poor data efficiency of RL methods, by al-

lowing policies learned in simulation to be more reliably deployed in the real world. The addition of explicit adaptation modules (Kumar et al., 2021), or the use of recursive policies, such as LSTM neural networks, has also contributed to the success of RL by allowing robots to observe information about the environment through proprioception. For example, Siekmann et al. (2021) learn a blind stair climbing controller for the Cassie biped, and Duan et al. (2022) use a similar policy to walk on constrained footholds in. With additional vision modules, they achieve perceptive locomotion over boxy terrain as well (Duan et al., 2023). Because reinforcement learning can struggle in scenarios with sparse footholds, Jenelten et al. (2024) proposed a hierarchical approach where an MPC footstep planner guides a lower level learned tracking policy for quadrupedal locomotion. Yu et al. (2024) propose the opposite, where reinforcement learning is used learn high level strategies like gait selection and foot placement, and MPC is used to generate and stabilize corresponding full body motions. This strategy is now controlling Boston Dynamics' Spot quadruped in industrial use cases (Boston Dynamics). While reinforcement learning is not a focus of this thesis, we propose a footstep planner and a safe-terrain segmentation module which could easily be used in one of these hierarchical frameworks.

3.1.5. Underactuated walking on Discontinuous Terrains

The development of controllers specifically for underactuated walking traces back to the development of passive dynamic walkers by McGeer (1990). The primary focus was on finding stable limit cycles for simple hybrid, nonlinear walking models (Garcia et al., 1998). Moving toward considering rough terrain, Ernst et al. (2009) developed a deadbeat feedforward controller for the SLIP model on uneven terrain, and Dai and Tedrake (2012) introduced a method for optimizing limit cycles for general hybrid walking models with an additional metric for robustness to ground height variation. The focus on limit cycle optimization posed a challenge for underactuated walking over stepping stones, as new motions need to be generated online to respect the foothold constraints. Several methods have therefore been proposed specifically for underactuated walking over stepping stones. With two notable exceptions (discussed at the end of this subsection), these works assume that the controller is provided a one-step preview of the terrain, and must use a mechanism other than foot placement to maintain a viable walking gait while stepping to the target foothold. Nguyen et al.

(2020) develop a gait library for planar walking walking over gaps of varying lengths, however the post impact state was often too far from the initial state of the optimized gait for the next gap, causing failure. Dai et al. (2022) use vertical CoM velocity as an alternative to foot placement for underactuated planar walking, while Xiang et al. (2024) use step timing to handle foot placement variation for 3D underactuated walking. Nguyen et al. (2016) propose using control barrier functions to adapt a nominal HZD based controller to respect foothold constraints online. In contrast to these works, the controllers developed in this thesis simultaneously optimize over the robot's dynamics and the choice of stepping surface for each step, enabling freeform walking over discontinuous terrain. The mixed-integer approach we take was previously assumed to be too slow for real-time deployment, however we overcome the speed concerns by employing a state of the art mixed-integer solver and focusing on the short-horizon task of tracking a desired velocity. Two other real-time controllers which optimize jointly over stepping surface and robot dynamics were published concurrently or shortly after our initial work on this subject (Acosta and Posa, 2023). Gu et al. (2024) formulate a model-predictive controller over ALIP dynamics which optimizes a signal-temporal-logic robustness metric, including distance to the edge of stepping stones as one criteria for robustness. Shim et al. (2023) incorporate an artificial potential field which enforces that the solution to a whole-body MPC problem respect polygonal foothold constraints. Because these methods are both based on nonlinear MPC, they yield locally optimal solutions, making it unclear how well they will translate to more challenging terrains or poor initializations.

3.2. Footstep Planning on Discontinuous Terrain

This section focuses on methods for footstep planning over rough terrain. The literature on safe bipedal footstep planning mainly considers humanoid robots with large feet, which allow a feasible center of mass trajectory to be planned and tracked (e.g. via LQR (Tedrake et al., 2015)) for any reasonable footstep plan. A benefit of these methods is that long-horizon plans can be generated quickly via traditional motion planning approaches like graph search (Chestnutt et al., 2003) by discretizing the free space into cells. However, because footsteps are not re-planned at high rates, decoupled approaches have slow walking speeds to avoid violating zero-moment-point constraints (Kajita et al., 2003).

Many features have been proposed for graph-search based footstep planners such as adaptive levels of detail (Hornung and Bennewitz, 2012) and the ability to leverage partial footholds (Griffin et al., 2019). For the rest of this section, though, we will focus on the use of mixed-integer programming for footstep planning. Mixed-integer programs are a class of mathematical optimization problem where a subset of the variables are constrained to take integer values. This formulation is attractive for our purposes, as mixed-integer constraints can be readily incorporated into existing model-predictive-control formulations for dynamic walking, which we will detail in Chapter 5.

3.2.1. Mixed Integer Footstep Planning

Deits and Tedrake (2014) introduced the use of mixed-integer convex programming for footstep planning by decomposing safe terrain into a collection of convex polygons, and using binary variables to assign every footstep to a polygon. This problem has a worst-case computational complexity of M^N, where M is the number of footholds and N is the planning horizon. Due to this combinatorial complexity, some efforts have been made to find convex relaxations and more efficiently solveable formulations. Tonneau et al. (2020) provide a convex approximation of this problem as a linear program, and Song et al. (2021) show how both the mixed integer and linear programming formulations can be made more efficient by using a simplified trajectory planner to prune irrelevant footholds. Marcucci et al. (2024) introduce shortest path problems over graphs of convex sets as a problem statement with a much stronger mixed-integer formulation (that is, the convex relaxation of the problem is much more informative about the true optimal solution) than previous approaches. In contrast to this thesis, these works focus on long horizon footstep planning, and only consider geometric criteria such as workspace constraints, and quasistatic stability criteria, such as the existence of a feasible center of mass trajectory which lies completely above the support polygon.

Mixed-Integer footstep planning has also been used for motion planning quadruped robots. Risbourg et al. (2022) use the convex relaxation from (Tonneau et al., 2020) online to project the desired footstep sequence to the closest convex footholds, subject to kinematic constraints. Corbères et al. (2023) incorporate this footstep planning strategy as an online foothold scheduler at 1-5 Hz with vision in the

loop. Due to the low planning rate, and the lack of dynamics constraints in the contact scheduler, they rely on a separate swing foot planner and whole body MPC to find feasible robot trajectories. Aceituno-Cabezas et al. (2018) formulate a full quadruped trajectory optimization problem using mixed integer constraints for assigning footsteps to footholds and to approximate the nonlinear manifold constraint for 3D rotations. Their trajectory optimization features both kinematic and dynamics constraints, but does not re-plan the footholds in real time. In this thesis, we shift the focus from offline, geometric optimizations, to real-time optimization which considers the robot's dynamics to stabilize dynamic walking over uneven terrain.

3.3. Identifying Safe Terrain from Perception

A significant barrier to deploying mixed-integer footstep planning methods on hardware is the need for a convex planar polygon decomposition of the terrain around the robot. Because there is no sensor which can directly measure convex planar polygons, we must rely on processing data from range sensors such as RGB-D cameras or lidar. Some works attempt to directly extract planar polygons from individual point clouds (Bertrand et al., 2020) or depth images (Mishra et al., 2021), however this excludes occluded terrain from the safe terrain set, including the terrain directly underneath the robot.

To estimate the full state of the terrain, multiple sensor streams must be fused over time into a single representation. In order to perform this fusion compactly, the field has turned to 2.5D elevation maps (Fallon et al., 2015). In particular, the formulation developed by Fankhauser et al. (2018) is popular because it estimates the elevation relative to the robot, rather than in a world-fixed frame. This eliminates the need to register input point clouds against the existing map to account for state estimate drift of the robot. This method was parallelized for execution on the GPU by Miki et al. (2022b).

The popularity of elevation maps has lead to a proliferation of algorithms for extracting convex planar polygons from elevation maps via plane segmentation (Miki et al., 2022b; Fallon and Antone, 2019). However, these approaches segment each elevation map independently, leading to issues with temporal consistency (Corbères et al., 2023; Acosta and Posa, 2023), especially because elevation

mapping is vulnerable to artifacts from drift in the floating base position estimate (Wisth et al., 2023; Grandia et al., 2022). To generate temporally consistent polygon constraints in real-time, despite these challenges imposed by legged locomotion, Bin et al. (2024) develop a GPU accelerated semantic mapping framework which incrementally updates estimates of floating base drift, and the state of planar polygon terrain, through a sequence of depth images and camera poses. This approach has advantages for stair climbing, where the terrain is known to be planar and precise foot placement is required, but could struggle in outdoor environments where the ground is not perfectly flat.

On unstructured terrain, some works compute heuristic costs from the elevation map to guide planning. McCrory et al. (2023) encode various traversability costs into a graph search algorithm for humanoid footstep planning. Jenelten et al. (2022) add a nonconvex cost on the gradient of the elevation map at the planned stance foot locations in their MPC formulation for quadrupedal walking. These heuristic costs recognize that planar polygons are a modeling choice to support optimization based control, not a necessary condition for steppability. In this thesis, we will present a terrain segmentation approach adopts a similar philosophy by classifying elevation map cells as steppable or not without regard for global planarity. Unlike heuristic footstep costs, our approach still allows for global optimality in the footstep planning problem by transforming this temporally consistent classification into a stepping stone representation for MIQP footstep planning.

CHAPTER 4

Background

4.1. Cassie

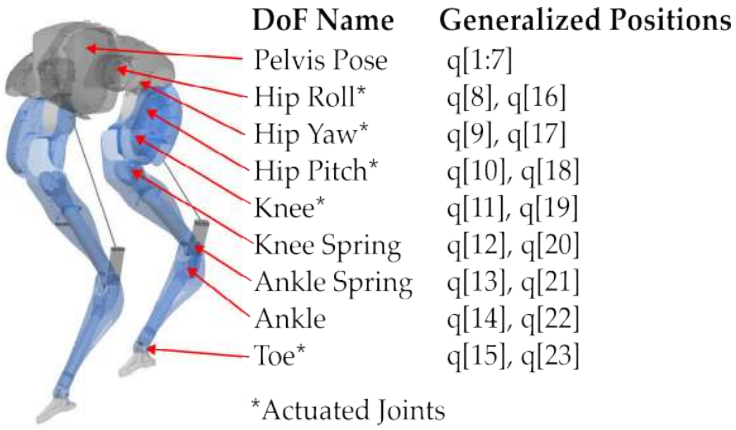


Figure 4.1: Cassie's configuration space.

The hardware platform used for this thesis is Cassie (Fig. 4.1), a 22 degree of freedom biped, manufactured by Agility Robotics. Ten of these degrees of freedom are actuated, four of them are constrained by stiff leaf springs, and another two are constrained by the mechanical linkage on each leg. The remaining 6 degrees of freedom represent the position and orientation of the floating base, which must be controlled through contact forces entering at the robot's feet. In single stance, Cassie's line foot only constrains five of the six degrees of freedom of the floating base, resulting in one underactuated degree of freedom. Often, the stance foot is assumed to have a passive ankle, resulting in an additional underactuated degree of freedom.

4.1.1. Rigid Body Cassie Model

This section introduces the equations of motion for Cassie, modeled as a collection of rigid bodies connected by revolute joints. We denote the configuration, or generalized positions, as $q \in \mathbb{R}^{n_q}$, where $n_q = 23$. The first four position coordinates represent the orientation of the robot, expressed as a quaternion representing the orientation of the pelvis frame in the world frame. The next three position coordinates represent the position of the pelvis frame in the world frame. The remaining

16 coordinates represent the joint angles for the revolute joints comprising the left leg, and then the right leg, including the deflection of the four leaf springs.

The generalized velocity, $v \in \mathbb{R}^{n_v}$, consists of the angular and linear velocities of the pelvis, measured and expressed in the world frame, and the angular velocities of the revolute joints. Because the angular velocity has one fewer component than the quaternion orientation, $n_v = 22$. Because the ankle spring joint does not have an encoder, we find its position via inverse kinematics, and set its velocity to zero.

The full state vector of the system is $x = [q, v]$, and the inputs to the system are joint torques, $u \in \mathbb{R}^{N_u}$, where $n_u = 10$. We note that we later use x and u to refer to the state and input variables of other models, omitting identifying subscripts for brevity, since the relevant system is clear from the surrounding context.

Dynamics

The map $N(q) \in \mathbb{R}^{n_q \times n_v}$ is used to convert v into $\dot{q}$, the time derivative of q:

$$\dot{q} = N(q)v. \tag{4.1}$$

$N(q)$ converts the angular velocity of the pelvis into the time derivative of the quaternion representing the pelvis orientation, and is an identity mapping for the remaining velocities.

The generalized accelerations, $\dot{v} \in \mathbb{R}^{n_v}$ can be derived from Lagrangian dynamics, taking the form of the manipulator equations

$$M(q)\dot{v} + C(q, v) = Bu + J_\lambda^T \lambda, \tag{4.2}$$

where $M(q) \in \mathbb{R}^{n_v \times n_v}$ is the configuration-dependent mass matrix, $C(q, v) \in \mathbb{R}^{n_v}$ are the Coriolis and gravity forces, $B \in \mathbb{R}^{n_v \times n_u}$ is the selection matrix mapping joint torques to generalized forces, $J_\lambda \in \mathbb{R}^{n_c \times n_v}$ is the constraint Jacobian (defined in the next subsection), and λ are the constraint forces. The constraint dimension n_c depends on the contact configuration.

Constrained Dynamics

Our floating base representation of Cassie contains only 10 actuators, but 22 velocities. With the exception of the underactuated degree of freedom about the line of contact in single-stance, these unactuated coordinates are controlled by constraint forces arising from the four bar linkages, springs[1] and ground contacts. These constraints are *holonomic constraints*, which can be written in the form $\phi(q) = 0$, where each row of $\phi : \mathbb{R}^{n_q} \mapsto \mathbb{R}^{n_c}$ eliminates a degree of freedom from the constrained dynamics. Differentiating, we have that $\dot{\phi}(q) = \frac{\partial \phi}{\partial q}\dot{q} = \frac{\partial \phi}{\partial q}N(q)v = 0$. The term $\frac{\partial \phi}{\partial q}N(q)$ is the "Jacobian with-respect-to-velocity" of the constraint, which maps generalized velocities to constraint velocities. We usually refer to this term as simply the *constraint Jacobian*, J_λ, and it is the same Jacobian which appears on the right hand side of (4.2). Differentiating again, we get an acceleration constraint,

$$J_\lambda \dot{v} + \dot{J}_\lambda v = 0. \tag{4.3}$$

This constraint is affine in $\dot{v}$, meaning that for a given robot state $x = [q, v]$ and input u, (4.2) and (4.3) form a set of linear equations which can be solved jointly for accelerations and constraint forces. The subset of constraint forces representing contact forces must additionally satisfy the *friction cone* constraint:

$$\mu \lambda_{c,n} \geq \sqrt{\lambda_{c,t_1} + \lambda_{c,t_2}} \tag{4.4}$$

where μ is the friction coefficient, $\lambda_c \in \mathbb{R}^3$ is a contact force, $\lambda_{c,n}$ is the normal component of the contact force and $\lambda_{c,t}$ are the tangential components. The term *friction cone* refers both to the constraint (4.4), as well as the set of forces which satisfy (4.4).

Hybrid Dynamics and Impacts

A feature of rigid-body dynamics models is that impact events at touchdown lead to discrete jumps in the robot's velocity via contact impulses which instantaneously bring the impacting foot to rest. In this thesis, we design controllers where the feet touchdown with zero velocity relative to the ground, eliminating these discrete jumps. Therefore, we update the contact model at touchdown

[1] In our Cassie simulations, we include actual revolute springs in the model, however for control, we make the simplifying assumption that the springs are fixed in their current positions

to correctly compute the contact Jacobian, but do not include impulsive reset maps in our control design.

4.2. Hierarchical Control of Bipedal Walking

Successfully stabilizing an underactuated system, such as a bipedal robot, requires reasoning about applying feedback to the actuated coordinates to affect the behavior of the whole system. When faced with such a dilemma, a typical route is to reach for optimal control (Tedrake, 2023), however real-time optimal control of the full nonlinear dynamics of legged robots has only become possible in recent years (Wensing et al., 2023), requiring extensive effort to formulate efficient numerical optimization algorithms and problem statements (Grandia et al., 2022; Mastalli et al., 2023).

Instead, roboticists have typically managed this complex task with hierarchical frameworks. A high-level layer formulates output trajectories as functions of the underactuated states, and a low-level layer controls the actuated degrees of freedom to track these outputs.

In frameworks like trajectory optimization and Hybrid-Zero-Dynamics, these output trajectories are synthesized offline. More commonly, however, controllers are synthesized using reduced-order models that capture the essence of the dynamics in a more tractable form. These dynamics are used online to find center of mass motions, contact forces, and footstep locations which stabilize the walking motion. The desired contact forces, center of mass dynamics, swing foot trajectory, and other relevant outputs are generally tracked with an inverse-dynamics based operational space controller (OSC) (Wensing and Orin, 2013a; Kim et al., 2019; Khatib, 1987). This OSC is often referred to as a "whole-body controller" due to its central role converting desired task-space dynamics to robot joint-torque commands. The rest of this section will discuss the reduced order models used for footstep planning in this work and the following section will discuss the operational space controller used for output tracking. Construction of the relevant outputs will be discussed in the appropriate chapters for each waking controller.

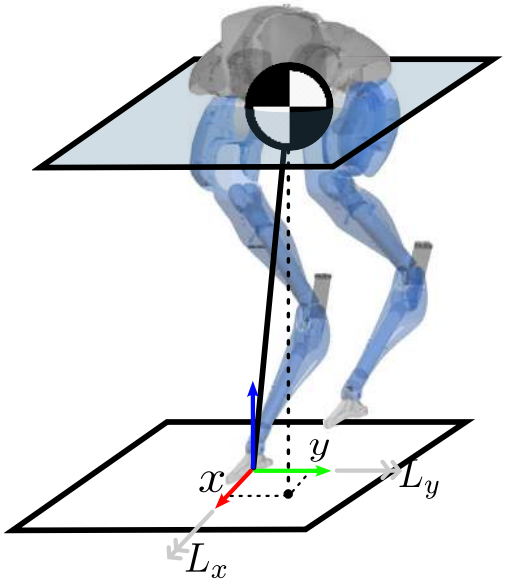


Figure 4.2: The ALIP model assumes that the robot's CoM is restricted to a virtual plane above the terrain. The states of the ALIP model are the horizontal CoM positions, and the angular momentum of the robot about the horizontal axes.

4.2.1. The ALIP model

The ALIP model (Fig. 4.2) is an approximation of the horizontal center-of-mass dynamics of the robot during single stance. The ALIP model makes the same assumptions of zero vertical CoM acceleration and zero angular momentum about the CoM as the Linear Inverted Pendulum (LIP) model (Kajita et al., 2001), but uses angular momentum about the contact point in place of center-of-mass velocity to describe the speed of the robot. Angular momentum about the contact point has the advantage of being relative-degree three to (non stance-ankle) motor torques, compared to relative-degree one for center-of-mass velocity (Gong and Grizzle, 2021). This makes the ALIP model less affected than the LIP by unmodeled dynamics of the full robot, such as extra horizontal ground-reaction-forces needed to re-circulate the swing leg.

The state of the ALIP model consists of the horizontal position of the center of mass relative to the stance foot, (x_{com}, y_{com}) and the tilting components of the angular momentum of the robot about the contact point (L_x, L_y). We direct the reader to (Gibson et al., 2022) for a full derivation of the ALIP dynamics starting from the centroidal dynamics of the robot and assuming piece-wise planar

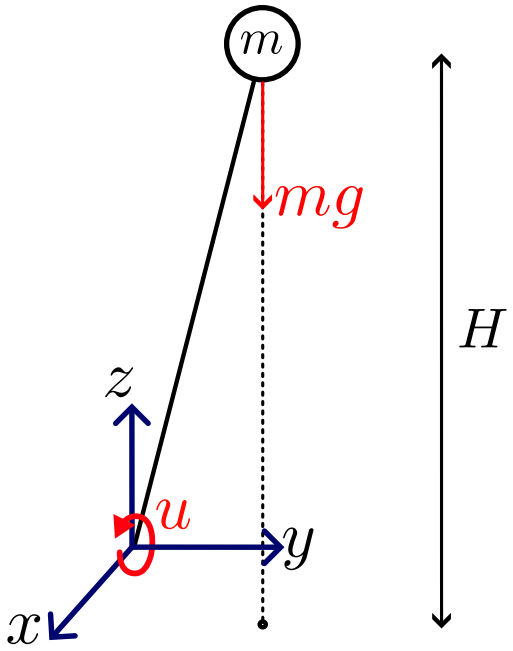


Figure 4.3: The ALIP model with mass m, constant CoM height H, and ankle torque in the sagittal plane, u.

terrain with a passive ankle. Here, we present a simplified derivation which neglects sloped terrain, and includes ankle torque in the sagittal plane, u, to take advantage of Cassie's blade foot (Fig. 5.3). In neglecting sloped terrain, we assume constant CoM height, so that $\dot{z}_{com} = 0$ in addition to the previously assumed $\ddot{z}_{com} = 0$. We consider forward walking to be in the $+x$ direction, so the sagittal plane is the $x - z$ plane and the coronal plane is the $y - z$ plane.

Let m be the mass of the robot, and H the height of the CoM above the terrain. We start with a moment balance about the horizontal axes,

$$\dot{L}_x = -mgy_{com} \tag{4.5}$$

$$\dot{L}_y = u + mgx_{com}. \tag{4.6}$$

Our assumption of constant CoM height ($\dot{z}_{com} = 0$) and zero centroidal angular momentum yield

$$L_x = -mH\dot{y}_{com} \tag{4.7}$$

$$L_y = mH\dot{x}_{com} \tag{4.8}$$

which can be rearranged as

$$\dot{x}_{com} = \frac{L_y}{mH} \tag{4.9}$$

$$\dot{y}_{com} = -\frac{L_x}{mH}. \tag{4.10}$$

Wrapping these equations up into state-space form, the single-stance dynamics of the ALIP with ankle torque are given by a continuous-time LTI system:

$$\underbrace{\begin{bmatrix} \dot{x}_{com} \\ \dot{y}_{com} \\ \dot{L}_x \\ \dot{L}_y \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \frac{1}{mH} \\ 0 & 0 & \frac{-1}{mH} & 0 \\ 0 & -mg & 0 & 0 \\ mg & 0 & 0 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_{com} \\ y_{com} \\ L_x \\ L_y \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}}_{B} u \tag{4.11}$$

It can be seen from (4.5) and (4.6) (which are valid for any point-foot walker) that the angular momentum about the contact point is relative-degree three to non-stance-ankle motor torques. $\dot{L}_x$ and $\dot{L}_y$ are functions of only the robot's configuration (via the CoM position) and ankle torque, requiring two more differentiations to reach an expression which includes the robot's other motor torques or the ground reaction forces. Therefore L_x and L_y are relative degree three to any reaction forces which disobey the assumptions of the ALIP model. In comparison, the CoM velocity differentiates to the CoM acceleration, which can be directly expressed in terms of the ground reaction forces.

4.2.2. Hybrid ALIP Model Based on Foot Placement

To enable control of the ALIP through foot placement, we derive a reset map relating the positions of the robot's feet at touchdown to a discrete jump in the ALIP state. Many walking controllers feature a double stance phase during which weight transfers from one leg to the other. A double stance phase is particularly useful for Cassie, to avoid oscillations caused by rapidly unloading Cassie's leaf springs. To treat the single and double stance phases as a single step in the step-to-step dynamics (Xiong and Ames, 2022), we derive a reset map from x_-, the ALIP state just before footfall, to x_+, the ALIP state just after liftoff, including a double stance phase of fixed duration, T_{ds}. We assume an impact-less touchdown, which does not cause a discontinuous jump in the ALIP state. This condition will be enforced for the full-order robot by planning swing-foot trajectories which touchdown with zero velocity. We start by integrating the double stance dynamics, and then we apply a coordinate change to express the ALIP state with respect to the new stance foot.

During double stance, we leave the ankles passive ($u = 0$) and treat the center of pressure (CoP) between the two ALIP point feet as a control input. We then integrate the resulting dynamics with an assumed input trajectory,

$$p_{CoP}(t) = p_- + f(t)(p_+ - p_-) \tag{4.12}$$

where $p_-, p_+ \in \mathbb{R}^3$ are the pre- and post-touchdown stance foot positions, t is the time since the beginning of double stance, and $f(t) : \mathbb{R} \mapsto [0, 1]$ determines the rate at which weight is transferred to the new stance foot.

The net ground reaction force, f_{grf} acts at the CoP and, under the assumption of zero centroidal angular momentum, points toward the CoM. Under the $\ddot{z}_{com} = 0$ assumption, this yields

$$f_{grf} = \begin{bmatrix} f_x & f_y & mg \end{bmatrix}^T \tag{4.13}$$

where

$$f_x = \frac{x_{com} - p_{CoP,x}}{H} mg$$

$$f_y = \frac{y_{com} - p_{CoP,y}}{H} mg.$$

Without loss of generality, we express the CoM and CoP positions as relative to p_-, then we sum moments about p_-, yielding

$$\dot{L} = p_{CoP} \times f_{grf} + \begin{bmatrix} x_{com} \\ y_{com} \\ H \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{4.14}$$

$$= \begin{bmatrix} mgP_{CoP,y} \\ -mgP_{CoP,x} \\ f_x P_{CoP,y} - f_y P_{CoP,x} \end{bmatrix} + \begin{bmatrix} -mgy_{com} \\ mgx_{com} \\ 0 \end{bmatrix} \tag{4.15}$$

Taking the horizontal components[2] of (4.15) and combining like terms gives

$$\dot{L}_x = -mg(y_{com} - p_{CoP,y}(t))$$

$$\dot{L}_y = mg(x_{com} - p_{CoP,x}(t)). \tag{4.16}$$

Substituting (4.12) into (4.16), we arrive at the continuous dynamics describing the ALIP during double stance:

$$\dot{x} = Ax + \underbrace{\begin{bmatrix} 0_{2\times1} & 0_{2\times1} & 0_{2\times1} \\ 0 & mg & 0 \\ -mg & 0 & 0 \end{bmatrix}}_{B_{CoP}} f(t)(p_+ - p_-). \tag{4.17}$$

[2]The reader may wonder if we can rightly neglect the impact of the third component, $\dot{L}_z = f_x P_{CoP,y} - f_y P_{CoP,x}$, on the full robot's dynamics. This term represents the 2D cross product between a vector along a line from p_- to p_+ and the vector from a point on this line to the projected CoM position. Given that we expect the center of mass to be close to above the line connecting centers of the two feet during double stance, we would expect the this term to be small enough for its effects to be absorbed by the whole-body controller as a disturbance.

The solution to (4.17) with the initial condition $x(0) = x_-$ is linear in x_-, p_- and p_+ (Ogata, 2001):

$$x(T_{ds}) = A_r x_- + B_{ds}\left(p_+ - p_-\right). \tag{4.18}$$

where $A_r = \exp(AT_{ds})$ and

$$B_{ds} = \left(\int_0^{T_{ds}} f(t) e^{A(T_{ds}-t)} dt\right) B_{CoP}. \tag{4.19}$$

For $f(t) = \frac{t}{T_{ds}}$, i.e. linearly shifting the robot's weight between feet over double stance, (4.19) evaluates to

$$B_{ds} = A_r A^{-1}\left(\frac{1}{T_{ds}} A^{-1}\left(I - A_r^{-1}\right) - A_r^{-1}\right) B_{CoP}. \tag{4.20}$$

In order to complete the reset map, we must transfer the ALIP state at the end of double-stance to the new stance foot. The angular momentum transfer formula is

$$L_+ - L(T_{ds}) = m v_{com}(T_{ds}) \times (p_+ - p_-). \tag{4.21}$$

Under the assumption of flat terrain and constant CoM height, the right hand side of (4.21) is the cross-product of two vectors in the $x - y$ plane, so the horizontal components are zero. Therefore $L_{xy+} = L_{xy}(T_{ds})$, so the remainder of the reset map is a coordinate change to express the CoM position as relative to the new stance foot:

$$x_+ = x(T_{ds}) + \underbrace{\begin{bmatrix} -I_{2\times 2} & 0_{2\times 1} \\ 0_{2\times 2} & 0_{2\times 1} \end{bmatrix}}_{B_{fp}} \left(p_+ - p_-\right), \tag{4.22}$$

with the fp subscript denoting "foot placement".

By sequentially applying (4.18) then (4.22), we arrive at a reset map from x_- to x_+ which is linear in x_-, x_+, p_- and p_+,

$$x_+ = \begin{bmatrix} A_r & (-B_{ds} - B_{fp}) & \underbrace{(B_{ds} + B_{fp})}_{B_r} \end{bmatrix} \begin{bmatrix} x_- \\ p_- \\ p_+ \end{bmatrix}. \tag{4.23}$$

4.2.3. Step-to-Step ALIP Dynamics

We will also consider step-to-step (s2s) ALIP dynamics. We view the ALIP without ankle actuation as a discrete-time linear time-invariant system by sampling the ALIP state at the end of each (fixed duration of T_{ss}) single stance phase. These dynamics are simply

$$x_{n+1} = A_{s2s} x_n + B_{s2s}(p_{n+1} - p_n) \tag{4.24}$$

where $A_{s2s} = \exp(A(T_{ss} + T_{ds}))$ and $B_{s2s} = \exp(AT_{ss})B_r$.

4.2.4. Step Timing Adaptation

We will use the fact that the initial ALIP state is constant to adapt the duration of the initial swing phase as part of the MPFC problem formulation. This has previously been applied to controllers based on the divergent component of motion (Englsberger et al., 2013; Khadiv et al., 2020; Xiang et al., 2024) and instantaneous capture point (Griffin et al., 2023), as these models admit an exact coordinate transform for the initial stance duration to make the touchdown state linear in the transformed variable. The ALIP state space does not admit this coordinate change, so we instead linearize the solution to (4.11). Given Given T seconds remaining in single stance, and an initial ALIP state x_c, the exact solution to (4.11) with constant ankle torque, u, is

$$x(T) = A_d(T)x_c + B_d(T)u \tag{4.25}$$

Where $A_d(T) = \exp(AT)$ and $B_d(T) = A^{-1}(A_d(T) - I)B$. We linearize (4.25) with respect to T and u about a nominal remaining stance time of T^* and ankle torque of 0 to find the ALIP state

at the end of the current stance period (and the initial state of the s2s ALIP model), x_0:

$$x_0 = A_d(T^*)x_c + \left.\frac{\partial A_d}{\partial T}\right|_{T^*}(T - T^*)x_c + B_d(T^*)u. \tag{4.26}$$

4.3. Operational Space Control

We use operational-space control (OSC) to track outputs such as swing foot position and pelvis orientation, while respecting frictional contact constraints (Wensing and Orin, 2013a). OSC performs input-output linearization based on the full-order Lagrangian model of the robot's dynamics (4.2).

We assume a set of outputs, $\{y_i(q)\}$, representing kinematics functions such as the center of mass position, swing foot position, or pelvis orientation, and a set of desired trajectories for these outputs to follow, $\{y_{i,des}(t)\}$. We also refer to these outputs as tasks, with the image of $y(q)$ being the *task space*. The task-space acceleration, $\ddot{y}_i$, is found similarly to the constraint acceleration (4.3):

$$\ddot{y}_i = J_i\dot{v} + \dot{J}_i v, \tag{4.27}$$

where $J_i = \frac{\partial y}{\partial q}N(q)$.

To track the desired output trajectories, we define task-space PD controllers,

$$\ddot{y}_{i,cmd} = \ ddoty_{i,des} + K_p(y_{i,des} - y_i) + K_d(\dot{y}_{i,des} - \dot{y}_i).$$

The goal of OSC is to find dynamically feasible inputs, generalized accelerations, contact forces, and constraint forces, such that the task-space accelerations, $\ddot{y}_i$, match the dynamics of the PD controller as closely as possible, while satisfying contact constraints and holonomic constraints. We

formulate this as a quadratic program with Lorentz cone constraints on the contact forces:

$$\underset{\dot{v},u,\lambda_h,\lambda_c,\varepsilon}{\text{minimize}} \sum_i^N \widetilde{\ddot{y}}_i^T W_i \widetilde{\ddot{y}}_i + \|u\|_W^2 + \|\dot{v}\|_W^2 + \|\varepsilon\|_W^2 \tag{4.28a}$$

$$\text{subject to } M\dot{v} + C = Bu + J_h^T \lambda_h + J_c^T \lambda_c \tag{4.28b}$$

$$J_h \dot{v} = -\dot{J}_h v \tag{4.28c}$$

$$J_c \dot{v} + \varepsilon = -\dot{J}_c v \tag{4.28d}$$

$$\lambda_c \in \mathcal{F} \tag{4.28e}$$

$$u_{min} \leq u \leq u_{max} \tag{4.28f}$$

where λ_c and J_c are the stacked contact forces and contact Jacobians, and $\mathcal{F}$ is the product of the friction cones for each contact point. The contact constraint is treated as a soft constraint by the introduction of a slack variable ε to ensure the problem is always feasible. The holonomic constraint $J_h \dot{v} = -\dot{J}_h v$ represents Cassie's four-bar linkages and fixed joint constraints to model Cassie's leaf spring springs. The task-space acceleration errors are $\widetilde{\ddot{y}}_i = \ddot{y}_{cmd} - (J_{y,i}\dot{v} + \dot{J}_{y,i} v)$.

4.4. Model Predictive Control

This thesis heavily leverages model predictive control (MPC), an optimal control method where feedback is applied to a system by recursively solving a finite-horizon trajectory optimization problem. MPC is a popular method for controlling robotic systems - especially legged robots - due to the ability to incorporate physical constraints such as joint and friction limits into the optimization problem. This section provide a summary of MPC as it pertains to this thesis. A more tutorial treatment of optimal control for robotics is given by Tedrake (2023), and a survey of the current state-of-the-art and challenges of MPC for legged robots is given by Wensing et al. (2023).

4.4.1. Linear-Quadratic MPC

Perhaps the simplest and most wide-spread form of MPC involves optimizing a convex quadratic cost on the state and input trajectory for of a discrete-time linear system over N time-steps (García et al., 1989). At each iteration, the controller solves (4.29) with the current measured state $\hat{x}$, and then

applies the first input from the solution, u_0 to the system, re-solving for a new u_0 at the next control step.

$$\underset{\{x_k\},\{u_k\}}{\text{minimize}} \quad x_N^T Q_N x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \tag{4.29a}$$

$$\text{subject to} \quad x_0 = \hat{x} \tag{4.29b}$$

$$x_{k+1} = A_k x_k + B_k u_k \tag{4.29c}$$

$$u_{min} \leq u_k \leq u_{max} \tag{4.29d}$$

$$C_k x_k \leq b_k, \tag{4.29e}$$

In (4.29), Q_N, Q, and R are positive semi-definite cost matrices, A_k and B_k define the system dynamics, u_{min} and u_{max} define the input limits, and C_k and b_k define path constraints on the state variables. This problem is a quadratic program (QP), and can usually be solved quickly using off the shelf QP solvers. By defining the dynamics as time-varying, we also allow for nonlinear MPC by linearizing the system's dynamics about a reference trajectory and solving the resulting convex QP. Repeated linear-quadratic approximation of a nonlinear MPC problem forms the basis for most of the real-time nonlinear MPC approaches used to control legged robots.

4.4.2. Nonlinear MPC for Legged Robots

Legged robots obey continuous, nonlinear, hybrid dynamics. Optimizing over these dynamics could be described by the following trajectory optimization problem, where x is the state, and ν are the

inputs:

$$\min_{x(t),\nu(t)} \Phi(x(T)) + \int_0^T L(x(t),\nu(t),t)\,dt \tag{4.30a}$$

$$\text{subject to } \dot{x}(t) = f(x(t),\nu(t)) \tag{4.30b}$$

$$x(0) = x_0 \tag{4.30c}$$

$$h(x,\nu,t) \geq 0 \tag{4.30d}$$

$$g(x,\nu,t) = 0. \tag{4.30e}$$

$$\tag{4.30f}$$

Usually, the order of hybrid transition events is pre-specified, with additional constraints relating the pre- and post- transition states. Almost universally, (4.30) is transcribed as a finite-dimensional nonlinear program using multiple shooting (Bock and Plitt, 1984), direct collocation (Hargraves and Paris, 1987) or differential dynamic programming (DDP) (Jacobson and Mayne, 1970). DDP is unique among these approaches in that it refers to both the problem transcription and the approach for solving it. DDP and related approaches like iLQR (Li and Todorov, 2004) are generally faster to solve to local optimality than either direct collocation or multiple shooting, but rely on a high-quality initial guess, and cannot handle constraints on the states or inputs. Generalizations have been proposed to handle input limits (Tassa et al., 2014) and more general path constraints on the state variables (Howell et al., 2019).

For real-time nonlinear MPC, multiple-shooting and collocation based approaches are generally solved using sequential quadratic programming (SQP) (Boggs and Tolle, 1995). SQP refers to the method of solving a nonlinear program via successive quadratic approximations of the cost, and linear approximations of the constraints (Nocedal and Wright, 2006). However, in the context on nonlinear MCP, SQP also refers to the practice of only solving one such QP per control iteration (Diehl et al., 2005), allowing successive control iterations to further refine the solution.

In order to achieve real-time performance, the dynamics for (4.30) have previously been defined via reduced order model dynamics (Jenelten et al., 2022) or centroidal dynamics (Romualdi et al.,

2022). Improvements in solvers and CPU speeds are increasingly allowing real-time MPC over kinodynamic models, which incorporate centroidal dynamics and joint kinematics, (Sleiman et al., 2021) and full Lagrangian dynamics models with joint torques (Mastalli et al., 2023; Grandia et al., 2022; Khazoom et al., 2024).

4.5. Robot-Centric Elevation Mapping

Our perception stack uses the robot-centric elevation mapping framework developed by Fankhauser et al. (2018) to construct a robot-centric elevation map of the terrain. This framework represents the terrain as a grid around the robot, with the height of each cell updated by point cloud measurements through a Kalman filter. The dynamics model for this Kalman filter defines a static map in the world frame. The measurement model is that 'incoming point clouds directly measure the height of the terrain. The point cloud points corresponding to each grid cell are accumulated and assigned a measurement noise based on a sensor-specific noise model. The height of each cell is then updated based on a one-dimensional Kalman filter. Letting $(\hat{h}, \sigma_h^2)$ be the estimated height and the variance of the height estimate, and $(\tilde{p}, \sigma_p^2)$ be the height measurement and height measurement variance, this update is given as

$$\hat{h}^+ = \frac{\sigma_p^2 \hat{h}^- + \sigma_h^{2-} \tilde{p}}{\sigma_p^2 + \sigma_h^{2-}}, \qquad \sigma_h^+ = \frac{\sigma_p^2 \sigma_p^2}{\sigma_p^2 + \sigma_p^2}. \tag{4.31}$$

We provide further details on our implementation of (Fankhauser et al., 2018) for Cassie in Chapter 7.

CHAPTER 5

Model Predictive Footstep Control

Parts of this chapter were previously published as parts of the following two publications:

Brian Acosta and Michael Posa. Bipedal Walking on Constrained Footholds with MPC Footstep Control. In *IEEE-RAS International Conference on Humanoid Robotics*, Austin, Texas, 2023.

Brian Acosta and Michael Posa. Perceptive Mixed-Integer Footstep Control for Underactuated Bipedal Walking on Rough Terrain. *arXiv preprint arXiv:2501.19391*, January 2025. Supplemental video for Chapters 5, 6, and 7: https://youtu.be/JK16KJXJxi4

This chapter presents Model Predictive Footstep Control (MPFC), a model-predictive-control-style footstep planner which allows underactuated bipeds to perform freeform walking over stepping stones. In addition to discrete foothold selection, MPFC optimizes over the continuous footstep positions, center of mass trajectory, ankle torque, and gait timing. MPFC is one of the first controllers to simultaneously optimize over the discrete choice of stepping surface and the robot's dynamics in real time[3], and to our knowledge, the research papers upon which this chapter is based represent the first deployment of such a controller on hardware.

We use binary variables to assign each footstep to a convex foothold (Deits and Tedrake, 2014), providing a straightforward extension of linear-quadratic MPC footstep controllers (Gibson et al., 2022) to discontinuous terrain, with the consequence that optimal control problem graduates in difficulty from a Quadratic Program to a Mixed-Integer-Quadratic Program (MIQP). MIQPs have been used extensively for offline trajectory optimization over broken terrains (Aceituno-Cabezas et al., 2018; Fey et al., 2024; Ding et al., 2020), but due to their combinatorial complexity in the planning horizon, they have seen much less use in real-time control. Our controller achieves solve times of

[3](Shim et al., 2023) was published concurrently with the conference paper associated with this chapter (Acosta and Posa, 2023) and uses artificial potentials to snap footsteps onto nearby footholds, and (Gu et al., 2024) was published shortly after (Acosta and Posa, 2023), and enforces stepping stone constraints with offline-generated signal-temporal-logic objectives.

less than 10 milliseconds by using a low dimensional, linear dynamics model, planning over a short footstep horizon, and eliminating foothold candidates far from the robot.

The remainder of the chapter is organized as follows. First, we transcribe MPFC as an MIQP, detailing the costs and constraints and the design rationale behind them. Next, we design outputs for realizing the MPFC solution on the Cassie. We then present results supporting MPFC's ability to stabilize walking over challenging terrains in real time. We conclude with a discussion of the overall capabilities and limitations of the approach and possibilities for future work.

5.1. MPFC Problem Statement

The following section details the formulation of our model predictive footstep controller as an MIQP. For the current stance phase, MPFC optimizes the stance duration and ankle torque. In the subsequent stance phases, MPFC only affects the s2s ALIP state through foot placement, encouraging footstep choices which are robust to limited ankle actuation. Because double stance is incorporated into the s2s dynamics, MPFC merges the double and single stance phase together into a combined stance phase, which is treated as single-stance within the optimization. The MPFC stance phase begins at each touchdown event with a nominal remaining stance time of $T_{ss} + T_{ds}$, and the footstep and gait timing solutions are ignored until the time since touchdown exceeds T_{ds} (Fig. 5.2).

The continuous MPFC decision variables are the step-to step ALIP states, x_n, the foostep positions, p_n, a constant ankle torque during the initial stance phase, u, and the remaining duration of the current stance phase, T. Each footstep is constrained to lie in a steppable region, $\mathcal{P}_i$, represented as a 2D polygon embedded in 3D space. We introduce one binary variable per discrete foothold per stance phase, $\mu_{n,i}$, corresponding to whether $p_n \in \mathcal{P}_i$. A diagram of the key MPFC decision variables is show in Fig. 5.1. We now introduce the MPFC problem statement (5.1), and then elaborate on the costs and constraints. Let x_c be the current ALIP state, with T^* seconds nominally remaining

in the current MPFC stance phase. MPFC is formulated as:

$$\underset{\mathbf{x},\mathbf{p},\boldsymbol{\mu},u,T}{\text{minimize}} \; J_{mpc}(\mathbf{x},\mathbf{p}) + J_{reg}(T,u) \tag{5.1a}$$

$$\text{subject to } x_0 = A_d x_c + AA_d x_c(T - T^*) + B_d u \tag{5.1b}$$

$$x_{n+1} = A_{s2s} x_n + B_{s2s}(p_{n+1} - p_n) \tag{5.1c}$$

$$\mu_{n,i} = 1 \implies p_n \in \mathcal{P}_i \tag{5.1d}$$

$$\sum_{i \in \mathcal{I}} \mu_{n,i} = 1 \tag{5.1e}$$

$$\mu_{n,i} \in \{0,1\} \tag{5.1f}$$

CoM, Input,Timing, and Footstep limits

5.1.1. Cost Design

Previous works use deviation from a reference ALIP trajectory as a state cost (Gibson et al., 2022), an approach taken in earlier versions of this work (Acosta and Posa, 2023). However, this implicitly encodes the corresponding footstep sequence into the state cost, and we desire for MPFC to freely pick the appropriate footstep sequence for a given terrain. Therefore we formulate a state cost which does not encode any particular gait. We penalize the distance of the MPFC solution from *the set* of ALIP trajectories which are periodic over 2 steps and achieve the desired velocity, v_{des}. This set is an affine subspace representing all possible x_n which satisfy the system (5.2).

$$(A_{s2s}^2 - I)x_n + A_{s2s}B_{s2s}\delta p_n + B_{s2s}\delta p_{n+1} = 0 \tag{5.2a}$$

$$\delta p_0 + \delta p_1 = 2T_{s2s}v_{des}, \tag{5.2b}$$

where $\delta p_n = p_{n+1} - p_n$, (5.2a) is the ALIP dynamics rolled out over two footsteps, with the period-2 orbit constraint $x_{n+2} = x_n$, and (5.2b) requires the net displacement of the robot to match the desired velocity. We show how to eliminate δp_n and δp_{n+1} in Section A.1.1, to express solutions

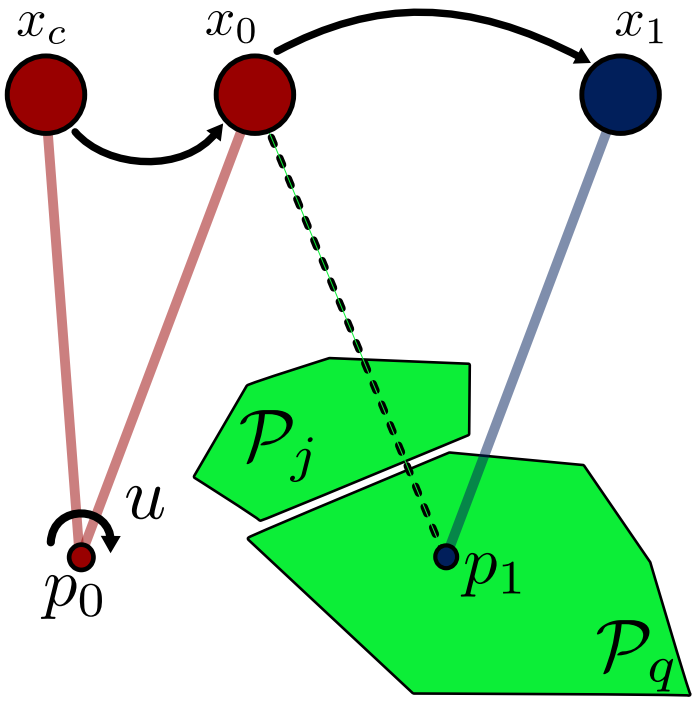


Figure 5.1: Key MPFC decision variables and constraints for a horizon of 2 stance phases. x_c is the current ALIP state, u is ankle torque applied during the current stance phase, x_0 is the ALIP state at the end of the current stance phase, and x_1 is the ALIP state at the end of the next stance phase. The current stance foot position, p_0, is unconstrained, and subsequent footsteps are constrained to lie in either $\mathcal{P}_j$ or $\mathcal{P}_q$ using integer variables.

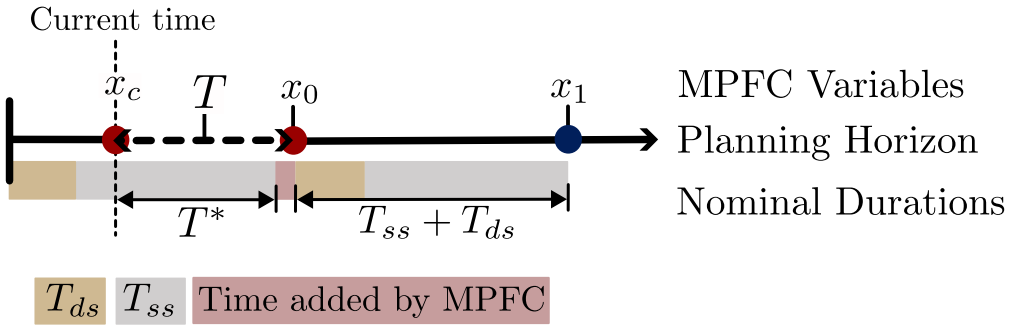


Figure 5.2: Relationship between the nominal stance phases and MPFC gait timing optimization. The initial stance duration is adjusted continuously by optimizing over the remaining stance time, T.

of (5.2) as

$$\Pi_n(x_n - d_n(v_{des})) = 0. \tag{5.3}$$

Where $\Pi_n \in \mathbb{R}^{4\times4}$ is a projection matrix used to eliminate δp_n from (5.2), and $d_n(v_{des})$ is an offset that encodes the desired velocity. Our MPC cost is then formulated as

$$\begin{aligned} J_{mpc}(\mathbf{x}, \mathbf{p}) = \sum_{n=1}^{N-1} \big[&(x_n - d_n)^T \Pi_n^T Q \Pi_n (x_n - d_n) + \\ &(\delta p_n - \delta p_n^*)^T R (\delta p_n - \delta p_n^*) \big] + \\ &(x_N - d_N)^T \Pi_N^T Q_N \Pi_N (x_N - d_N) \end{aligned}$$

where Q, R, and Q_N are positive-definite weight matrices. We regularize the relative footstep positions to a nominal step size, defined by the desired velocity and the step width, l, as

$$\delta p_n^* = \begin{bmatrix} v_{des,x}(T_{ss} + T_{ds}) \\ v_{des,y}(T_{ss} + T_{ds}) + \sigma_n l \\ 0 \end{bmatrix} \tag{5.4}$$

where $\sigma_n = -1$ for left-stance and $+1$ for right stance. We add quadratic costs on T and u, weighted by positive scalars w_T and w_u:

$$J_{reg} = w_T \|T - T^*\|^2 + w_u \|u\|^2. \tag{5.5}$$

5.1.2. Dynamics Constraints

The initial state constraint (5.1b) evaluates (4.26) to relate the current ALIP state to the initial s2s ALIP state via ankle torque and stance duration. The dynamics constraints (5.1c) are the s2s ALIP dynamics (4.24).

5.1.3. Foothold Constraints

Each convex polygonal foothold is defined by a plane $f_i^T p = b_i$ and a set of linear constraints $F_i p \leq c_i$. The logical constraint (5.1d) is enforced with the big-M formulation

$$F_i p_n \leq c_i + M(1 - \mu_{n,i}) \tag{5.6a}$$

$$f_i^T p_n \leq b_i + M(1 - \mu_{n,i}) \tag{5.6b}$$

$$-f_i^T p_n \leq -b_i + M(1 - \mu_{n,i}). \tag{5.6c}$$

With appropriately normalized F_i and f_i, (5.6) corresponds to relaxing each foothold constraint by M meters when $\mu_i = 0$. M must be large enough for every relaxed foothold to contain every unrelaxed foothold, but should otherwise be small for numerical stability. Since our problem scale is on the order of 2 m, we choose M = 10 for simplicity. The binary constraint (5.1f) and the summation constraint (5.1e) imply that exactly one foothold must be chosen per stance phase, and the remaining footholds must be relaxed.

5.1.4. CoM, Timing, Input, and Footstep Limits

We add the following constraints to reflect the physical limitations of the robot:

- We add a soft-constraint on the CoM position of ±35 cm. in each direction.
- We update bounds on T at each solve so the total single-stance duration lies in the range [0.27, 0.33] seconds.
- We add a crossover constraint to prevent the feet from crossing the $x - z$ plane.
- We limit the ankle torque to 22 Nm to keep the center of pressure within the blade foot.
- With $T_{min} = 0.27$ seconds left in the nominal single stance time, we add a trust region constraint on p_1. This constraint is a bounding box centered at the previous p_1 solution with a radius of T^* m. As implied by the unit conversion of T^* to a distance, the radius of this bounding box shrinks at a rate of 1 m/s.

5.2. Output tracking via Operational Space Control

To realize the planned walking motion on the physical robot, MPFC outputs are tracked with OSC (Fig. 7.1C). This section describes the construction of the outputs tracked by the OSC. The key contributions are a virtual constraint on the center of mass to enforce the ALIP assumption of piecewise planar terrain, and an adaptive-clearance swing-foot trajectory based on the displacement of the swing foot. Because the outputs are constructed only as functions of the current, previous, and upcoming stance foot locations, the footstep planner can be abstracted away from the OSC, and there is no need to interact with perception data inside of the high-rate OSC control loop.

5.2.1. Center of Mass Reference

Given a footstep plan, we construct a CoM trajectory which enforces the local planarity assumption of the ALIP model by constructing the least-inclined plane passing through the current and imminent stance foot positions (Fig. 5.3). Letting $p = p_{n+1} - p_n$, the plane parameters are the solution to

$$\begin{bmatrix} p_x & p_y \\ -p_y & p_x \end{bmatrix} \begin{bmatrix} k_x \\ k_y \end{bmatrix} = \begin{bmatrix} p_z \\ 0 \end{bmatrix}. \tag{5.7}$$

After solving for k_x and k_y, we define the reference trajectory for the CoM height in the stance frame as

$$z_c(t) = H + k_x x_c(t) + k_y y_c(t). \tag{5.8}$$

To account for discontinuities in $k_x, k_y, x_c,$ and y_c when the stance foot changes, we add a first-order low pass filter on k_x and k_y with a 100Hz cutoff frequency, and we clip the desired z_c to within 2.5 cm of the measured CoM height. Because $x_c(t)$ and $y_c(t)$ are updated to their current values, this reference trajectory is equivalent to using PD control to virtually constrain the CoM to the desired plane.

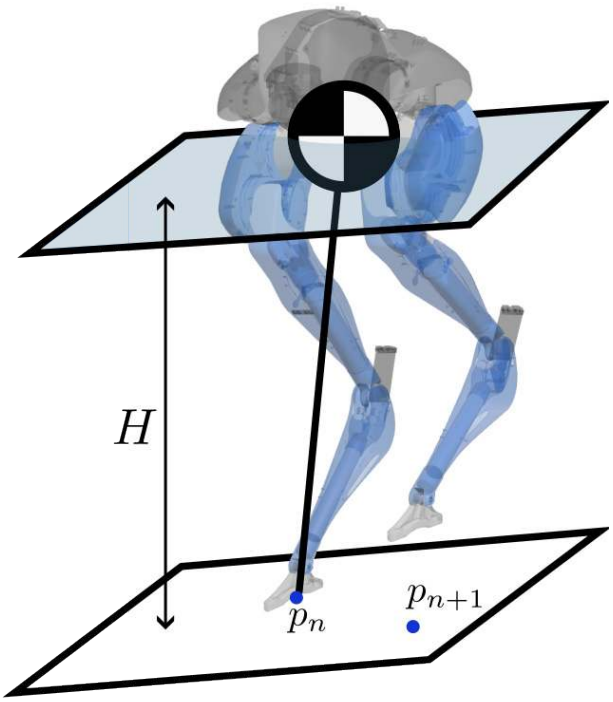


Figure 5.3: To enforce the planarity assumption of the ALIP, we use OSC to drive Cassie's CoM to a virtual plane defined by current and upcoming stance foot positions.

5.2.2. Swing Foot Reference

We continuously adapt the swing foot trajectory $p_{sw}(t)$ to the updated swing-phase duration and planned next footstep position with a planning QP similar to Khadiv et al. (2020). The planning QP adapts the previous trajectory so that the commanded position, velocity, and acceleration are continuous, while updating the trajectory midpoint, duration, and endpoint to match the desired swing-foot clearance and command from MPFC. First we generate a waypoint above the line connecting the initial and final foot location, following an adaptive clearance scheme, then we find a single-segment polynomial trajectory through this waypoint.

Adaptive Swing Foot Clearance

Our clearance scheme (Fig. 5.4) updates the midpoint of the swing-foot trajectory by adapting its direction and clearance to the total displacement of the swing foot. This gives high clearance when stepping up or down over steps without unnecessarily high steps on flat ground.

Let the swing foot position at the beginning of the swing phase be $p_{sw,0}$, the target foot position for the end of swing be $p_{sw,des}$, and define $\Delta p = p_{sw,des} - p_{sw,0}$. We construct a unit vector $\hat{n}_p$ which is perpendicular to Δp and lies in the plane spanned by Δp and the world z axis. When Δp is small, for example when the robot is stepping in place, small variations in height estimates can lead to $\hat{n}_p$ pointing in inconsistent directions, therefore we blend $\hat{n}_p$ with the unit z-vector, $\hat{e}_z$ to get a blended direction, $\hat{n}_b$:

$$\hat{n}_b = (1 - s)\hat{e}_z + s\hat{n}_p$$

where

$$s = \text{clamp}\left(\frac{\|\Delta p\| - 0.1}{0.1}, 0, 1\right).$$

The final waypoint location is then defined as

$$p_{mid} = p_{sw,0} + \frac{1}{2}\Delta p + c_{clear}\frac{\hat{n}_b}{\|\hat{n}_b\|},$$

where $c_{clear} = c + \min(c, \Delta p_z)$ is the final swing foot clearance, and c is a tuneable parameter representing the swing foot clearance on flat ground, which we set to 15 cm in our experiments.

Swing foot Planning QP

After finding the desired mid-spline waypoint p_{mid}, we solve (5.9) to update the swing foot trajectory to the new footstep target $p_{sw,des}$ and swing phase duration T. In addition to passing through the desired midpoint and ending at the target location, we constrain the swing foot trajectory to be continuous up to acceleration with the previously planned swing foot trajectory:

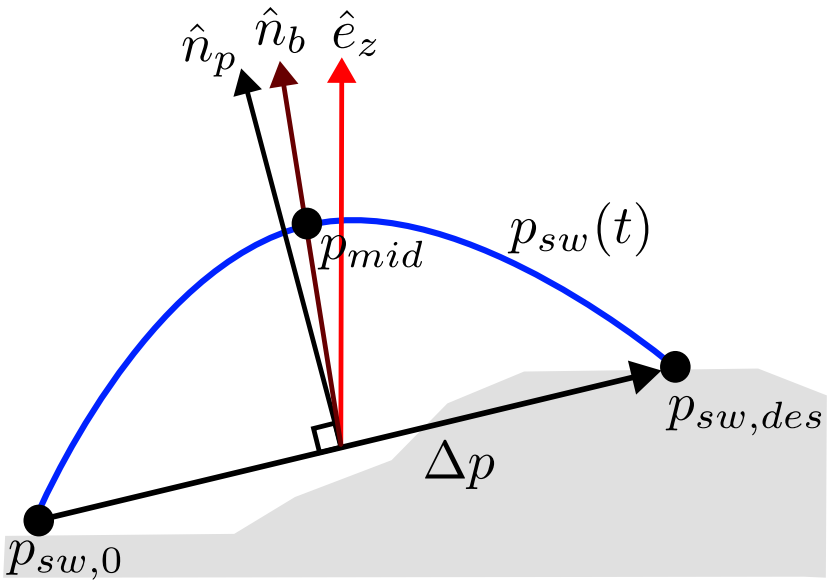


Figure 5.4: Trajectory from the swing foot position at the beginning of the swing phase, $p_{sw,0}$ to the next footstep solution from MPFC, $p_{sw,des}$. We adapt the direction and clearance of the trajectory's midpoint, p_{mid}, based on the relative positions of $p_{sw,0}$ and $p_{sw,des}$ to ensure sufficient ground clearance.

$$
\begin{aligned}
&\text{minimize} \int_0^T \ddot{p}_{sw}(t)^2 dt && && (5.9)\\
&\text{subject to } p_{sw,k}(t_{k-1}) = p_{sw,k-1}(t_{k-1}) && p_{sw}(T) = p_{sw,des}\\
&\qquad \dot{p}_{sw,k}(t_{k-1}) = \dot{p}_{sw,k-1}(t_{k-1}) && \dot{p}_{sw}(T) = 0\\
&\qquad \ddot{p}_{sw,k}(t_{k-1}) = \ddot{p}_{sw,k-1}(t_{k-1}) && \ddot{p}_{sw}(T) = 0\\
&\qquad p_{sw}(T/2) = p_{mid}
\end{aligned}
$$

where k indexes each OSC control cycle. We transcribe (5.9) as a QP which optimizes over the coefficients of a polynomial representing the swing foot trajectory. By using the initial swing foot position as $p_{sw,0}$ at the beginning of the swing phase, we ensure that the trajectory starts at the initial swing foot position without needing to explicitly enforce that constraint for every control cycle.

5.2.3. Other References

We track a constant pelvis roll and pitch of zero, and a constant swing-leg hip yaw (abduction) angle of zero. We track a commanded pelvis yaw rate from the remote control, and a swing toe angle so that Cassie's foot makes an angle of $\arctan k_x$ with the ground. We add a quadratic cost on the difference between MPFC and OSC ankle torque commands.

5.3. Results

This section presents results to support the ability of our locomotion framework to stabilize underactuated walking over discontinuous terrain in real time. First, we show simulation experiments validating the effectiveness of our framework over challenging terrains. In addition to the mixed-integer foothold constraints, our MPFC formulation includes a non-standard cost design, and the ability to optimize over the initial step duration. We perform ablations to validate the usefulness of these features. Then we show velocity tracking and solve time results from hardware experiments over varied terrain. We defer the explanation of the full perceptive locomotion hardware implementation to Chapter 7, presenting only the results relevant to MPFC here.

5.3.1. Simulation Results

This subsection details simulation experiments on complex terrains. Our simulation is written using Drake (Russ Tedrake and the Drake Development Team, 2019), and includes Cassie's leaf springs, reflected inertia, motor curves, joint limits, effort limits, and full collision geometry. We show the idealized capabilities of MPFC using ground-truth state and terrain information. We first present Cassie using our locomotion framework to walk over over an 8 m $\times$ 23 cm beam (Fig. 5.5), and up stairs with a depth of 27 cm and a rise of 15 cm (Fig. 5.6). MPFC is commanded a velocity of $[v_x, v_y] = [0.375, 0]$ m/s, and OSC is commanded a yaw rate proportional to the heading error. We show the velocity tracking of the controller over these terrains in Fig. 5.7.

Freeform walking

To emphasize that MPFC chooses the optimal foothold in real time, we show Cassie walking over constrained footholds with velocity commands from a human operator (Fig. 5.8).

Figure 5.5: Cassie walks across an 8 m × 23 cm beam using MPFC.

Figure 5.6: Cassie walks up a set of stairs with a 27 cm depth and a 15 cm rise using MPFC.

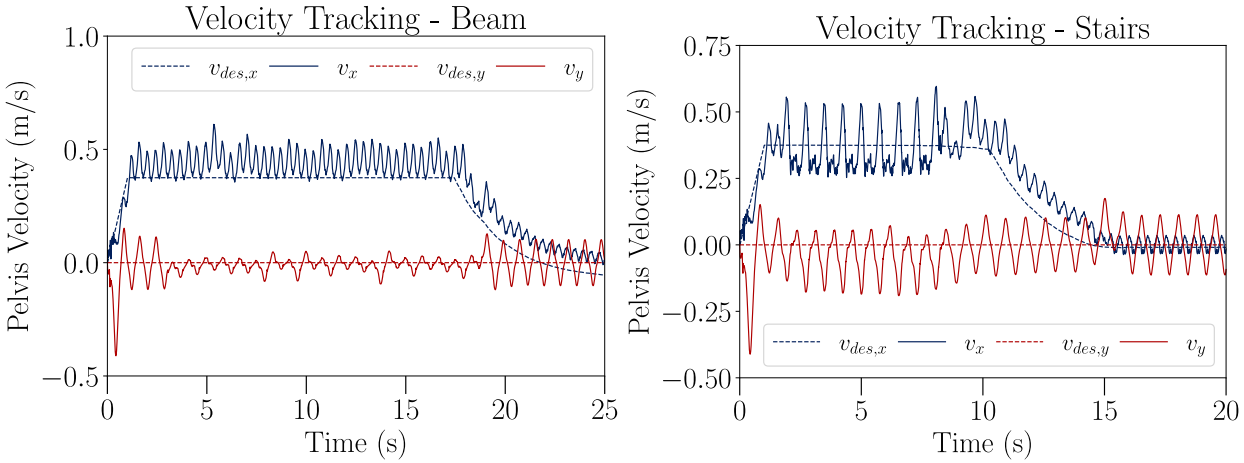


Figure 5.7: Velocity tracking for the beam and stairs results shown in Fig. 5.5 and Fig. 5.6

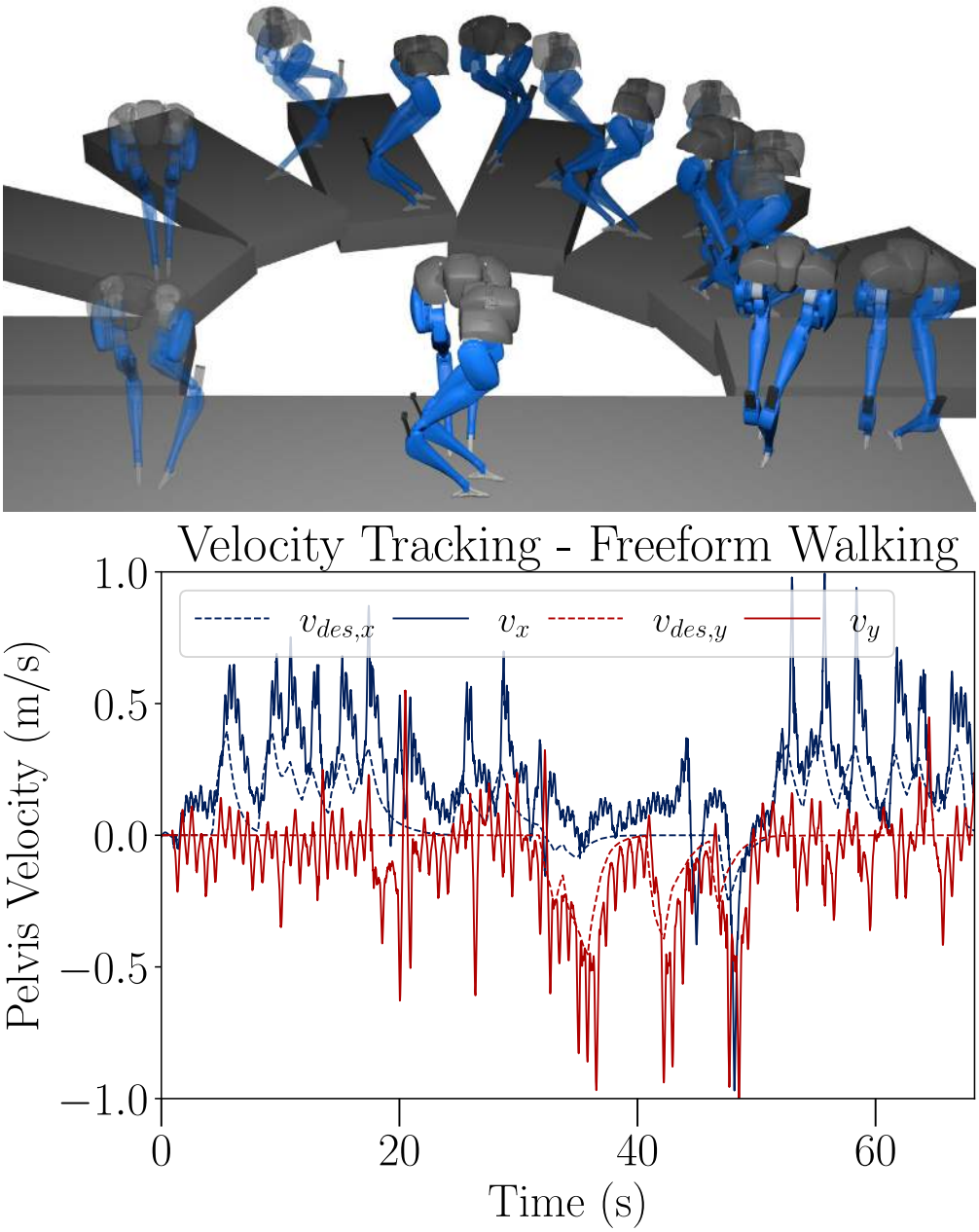


Figure 5.8: Freeform walking over constrained footholds using MPFC. **Top:** Cassie's pose throughout the freeform walking trial. **Bottom:** Velocity Tracking during the freeform walking trial.

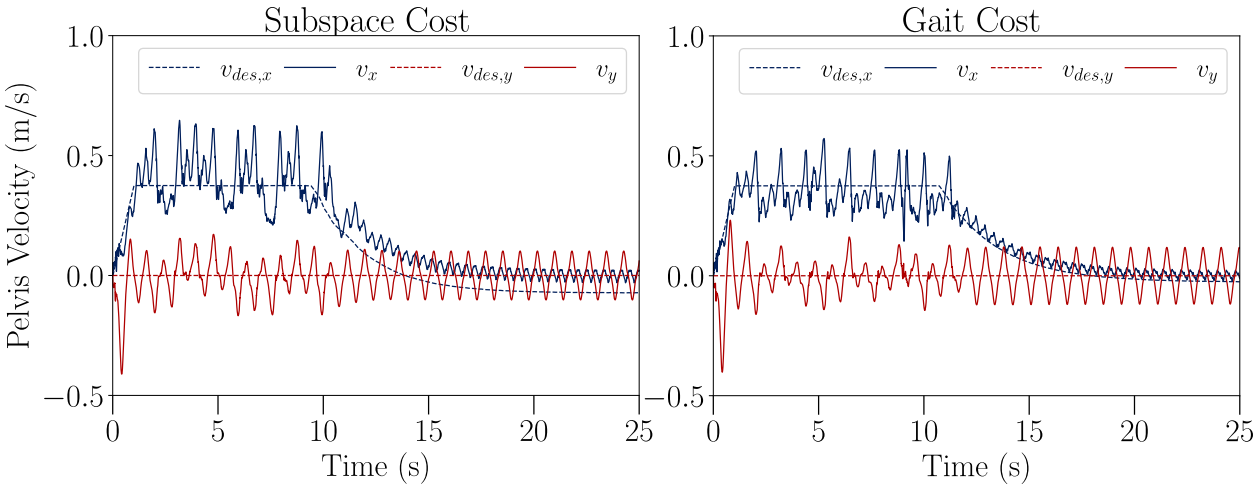


Figure 5.9: Comparing the velocity tracking performance of MPFC over 40 cm × 15 cm stairs using a naive cost and the subspace cost we develop in Section 5.1.1. The naive cost (Gait Cost) is on deviation from a reference trajectory, while our subspace cost (Subspace Cost) is on deviation from the subspace of Period-2 ALIP trajectories with the desired velocity.

Cost Comparison

We compare the performance of MPFC using the subspace cost and a gait-tracking cost based on the nominal stepping pattern. The gait tracking cost is formulated as

$$\sum_{i=0}^{N-1} \left((x_n - x_{d,n})^T Q (x_n - x_{d,n})\right) + (x_N - x_{d,N})^T Q_N (x_N - x_{d,N}) \tag{5.10}$$

where $x_{d,n}$ is the solution to (5.2a) after substituting (5.4) in for δp_n. This desired ALIP trajectory represents a single point within the desired velocity subspace, which makes a consistent amount of progress in both left and right stance. For both conditions, we use the same values of Q and Q_N, as well as regularization on gait timing, ankle torque, and relative footstep location. We walk up 40 cm × 15 cm stairs with both costs and show the velocity tracking performance in Fig. 5.9. The gait cost condition walks slightly slower than the commanded velocity, while the subspace cost walks slightly faster. When driving the physical robot with a remote control, this qualitatively feels like the subspace cost results in less hesitation before stepping over a ledge, making the robot easier to steer.

Step-Timing Optimization

We demonstrate the importance of step-timing optimization by measuring the success rate of walking across randomly generated stepping stones with and without step-timing optimization. We generate a 5×3 grid of stepping stones, where each stone has a random height, length, width, and position offset. The minimum length and width are controlled by the parameter d_{min}, and the centers

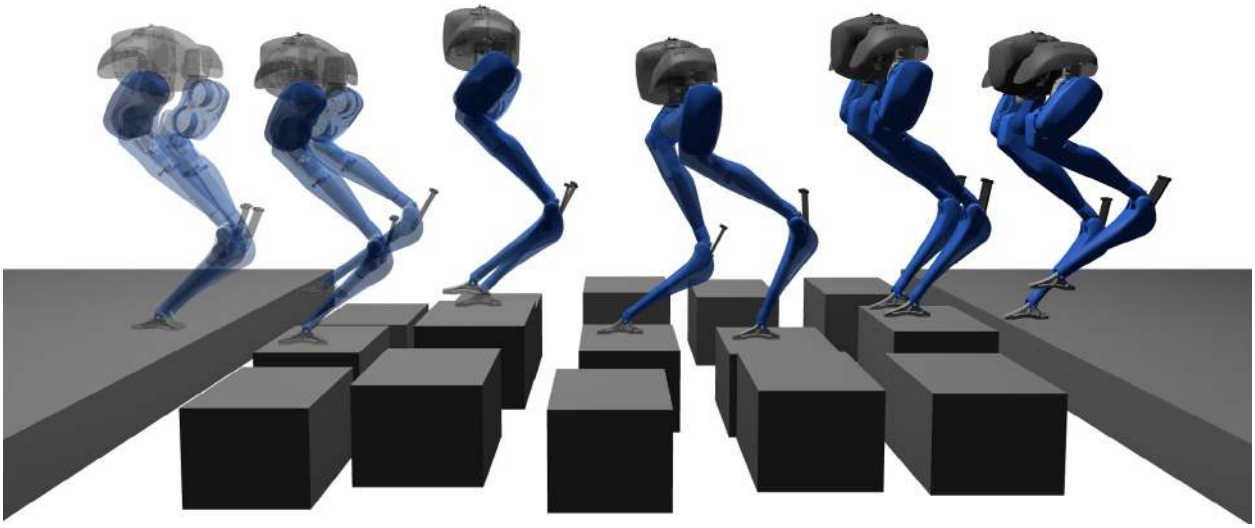


Figure 5.10: Example of successfully traversing a random stepping stone environment in simulation with $d_{min} = 35$ cm.

are offset from a nominal spacing of $d_{min} + 21$cm. The stepping stone dimensions are uniformly distributed with the bounds given in Table 5.1. An example stepping-stone terrain can be seen in Fig. 5.10. We sweep d_{min} from 35 cm to 70 cm and compute the success rate for traversing 50 random terrains at each size, which we report in Fig. 5.11.

Table 5.1: Stepping Stone Parameter Distributions

Parameter	**Uniform Distribution Bounds**
x offset	$\pm$ 5 cm
y offset	$\pm$ 5 cm
z offset	$\pm$ 7.5 cm
length	$[d_{min}, d_{min} + 5]$ cm
width	$[d_{min}, d_{min} + 5]$ cm

Step-timing optimization increases the success rate of walking over stepping stones, with larger

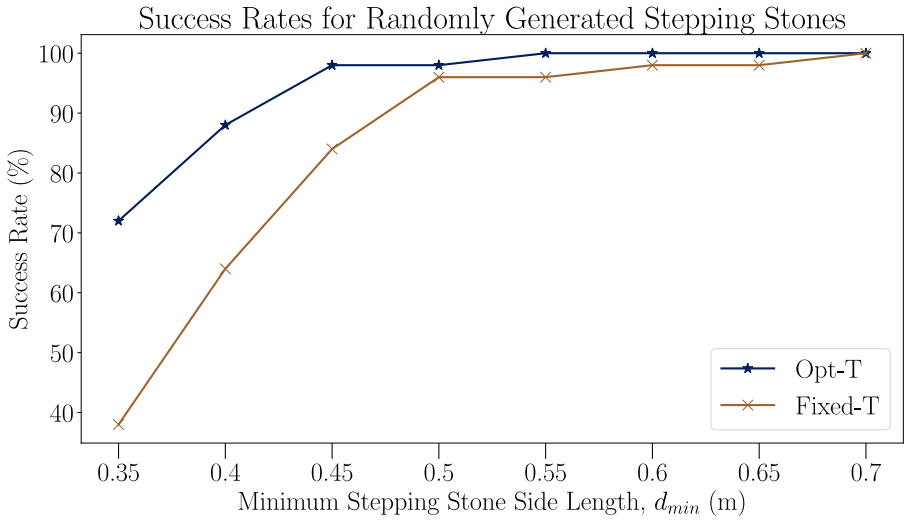


Figure 5.11: Success rates for walking across randomly-generated stepping stones in simulation. Results with step-timing optimization are labeled Opt-T, and results without step timing optimization are labeled Fixed-T. Step-timing optimization increases success rates over small footholds.

effects for terrains with smaller footholds. Intuitively, underactuated dynamics strongly couple step-timing, stride length, and walking speed. Given Cassie's underactuation, step-timing therefore compensates for variability in the foothold location.

5.3.2. Hardware Results

We present hardware results to support our claim of real-time footstep planning over rough terrain.

Walking on Discontinuous Terrains

A single trial traversing steps, a curb, and a grass hill is shown in Fig. 5.12. Additional trials are shown in the supplemental video.

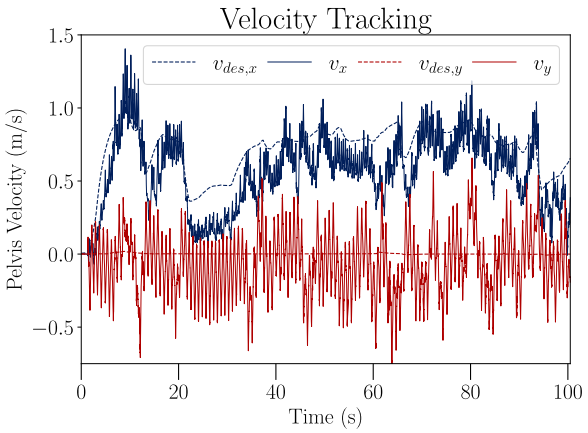
5.3.3. Controller Solve Times

To support our claims of faster than 100 Hz MPFC solve times, we compile solve times across 11:17 minutes of walking data from three experiments on the brick steps shown in Fig. 5.13. We give

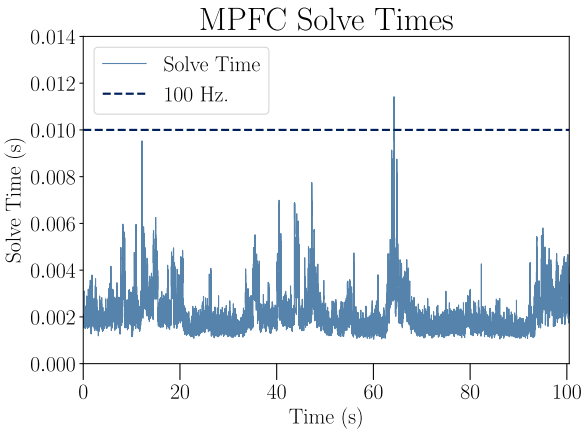
Table 5.2: MPFC Solve-Time Statistics (134,654 Solves)

Mean	Median	99.9th Percentile	Maximum
0.0022	0.0020	0.0077	0.0126

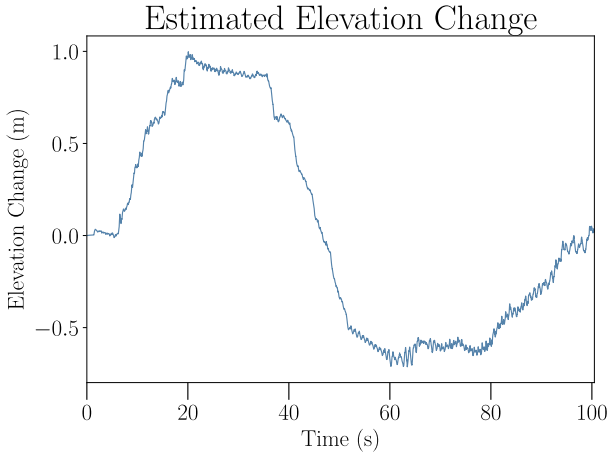
summary statistics of MPFC solve times in Table 5.2. This data uses a planning horizon of $N = 2$ footsteps, plus the initial single stance phase. The maximum solve time observed was 12.6 ms, with 99.9% of solves taking less than 7.7 ms.



(a) Plot of the velocity tracking performance of the robot using our control stack.



(b) MPFC Solve times during the trial in Fig. 5.13.



(c) Plot of the elevation change over the trial in Fig. 5.13, estimated from the onboard state estimator.

Figure 5.12: Velocity tracking, solve times, and elevation change for Cassie walking on discontinuous terrain including steps, a curb, and a grassy hill.

Figure 5.13: Motion tiles showing Cassie ascending and descending steps, stepping over a curb onto the grass, and walking up a grassy slope in one continuous walking trial using our proposed locomotion stack and the perception pipeline proposed in Chapter 6.

5.4. Conclusion

The following sections discuss limitations and future work and summarize the contributions of this chapter. We defer a detailed discussion of future work related to the full perception-integrated system to Chapter 7, after we have discussed the perception stack and full-stack walking experiments.

5.4.1. Limitations and Future Work

While the controller we present expands the capabilities of underactuated bipeds, some improvements could be made to tackle faster walking speeds and more difficult terrains. The planning horizon of 2 footsteps can result in overly optimistic footstep choices in antagonistic scenarios. Further research could focus on stronger mixed integer formulations or improved mixed-integer solvers to enable solving for longer footstep horizons in real time. Alternatively, additional robustness terms could be incorporated in MPFC to favor conservative behaviors. Cassie's small lateral workspace and MPFC's fixed stepping pattern make the controller vulnerable to lateral perturbations, especially those occurring at the beginning of single-stance. Robustness against these perturbations could be increased by including crossover steps, and by smaller minimum stance times. The stance duration should then be coupled to the swing-foot workspace to maintain reachability of the planned footsteps.

5.4.2. Summary

This chapter introduced Model Predictive Footstep Control, which synthesizes ALIP-based footstep control and mixed-integer footstep planning for underactuated walking over stepping stones. We increased the effectiveness of the controller by including ankle torque in the sagittal plane and step-timing optimization, essentially leveraging as much control authority as possible while maintaining the (mixed-integer) convexity of the MPC problem. We introduced a state cost which tracks to an affine subspace of the ALIP state-space based on the desired velocity, permitting more flexibility of the footstep choice than a trajectory-based cost. We developed a set of outputs for realizing the motions plans from MPFC using operational space control. We validated our controller and decision decisions through simulation experiments over complex terrains. We concluded with hardware experiments on Cassie, demonstrating the real-time nature of MPFC while traversing steps, curbs,

and a grassy hill. These hardware experiments were conducted with the aid of a real-time perception stack for generating stepping stone constraints from an RGBD camera, which we detail in the next chapter.

CHAPTER 6

Stable Steppability Segmentation and Convex Decomposition

Parts of this chapter were previously published as parts of Brian Acosta and Michael Posa. Perceptive Mixed-Integer Footstep Control for Underactuated Bipedal Walking on Rough Terrain. *arXiv preprint arXiv:2501.19391*, January 2025.

Supplemental video for Chapters 5, 6, and 7: https://youtu.be/JK16KJXJxi4

Our control framework for walking over convex polygons requires an effective pipeline for approximating the safe terrain as convex polygons online. This chapter introduces our solution, "Stable Steppability Segmentation" (S3) and a complementary convex decomposition procedure.

We argue that requiring a one-to-one correspondence between foothold constraints in the controller and real planar polygons in the environment (Grandia et al., 2022; Corbères et al., 2023) is overly restrictive and brittle. Our approach recognizes that planar polygons are a modeling choice used to support optimization-based control, rather than a hard safety requirement. By focusing on avoiding terrain which is clearly unsafe, we arrive at a simple algorithm which is robust to non-planar surfaces and more temporally consistent than explicit plane segmentation.

Our perception stack consists of two stages. The first stage, S3, uses local safety criteria and a simple hysteresis mechanism to classify elevation map pixels as safe or unsafe, resulting in a binary steppability mask. The second stage of our segmentation algorithm generates a set of convex polygons approximating the safe terrain identified by S3. We perform approximate convex decomposition (Lien and Amato, 2006), then take a convex inner-approximation of the resulting polygons before finally fitting plane parameters to these convex polygons using the original elevation map. While the S3 implementation in this thesis uses intuitive heuristic criteria for steppability classification, the general algorithm supports any number of criteria, allowing for composition with learning-based approaches and higher-level obstacle detectors.

In contrast to plane segmentation, we *do not* subdivide or reject any steppable region based on its estimated normal or its error with respect to a best-fit plane. This approach prevents localized frame-to-frame variations from having an outsized effect on the final segmentation, because there are no subdivision boundaries which might vary between frames, and localized outliers cannot trigger a subdivision or rejection of an entire region.

Because safety criteria are local, we further enhance temporal consistency by simply adding hysteresis to the classification of each pixel. We also use additional metrics beyond gradient or roughness to determine steppability, as discussed in Section 6.1. The simplicity of our approach allows the entire pipeline from elevation mapping to publishing convex polygons to run in real time on a single CPU thread.

In the remainder of this chapter, we first present the S3 and convex decomposition algorithms in detail, before presenting results to support S3's improved temporal consistency and computation time compared to plane segmentation. We conclude with opportunities for further development of S3 and a summary of out contributions.

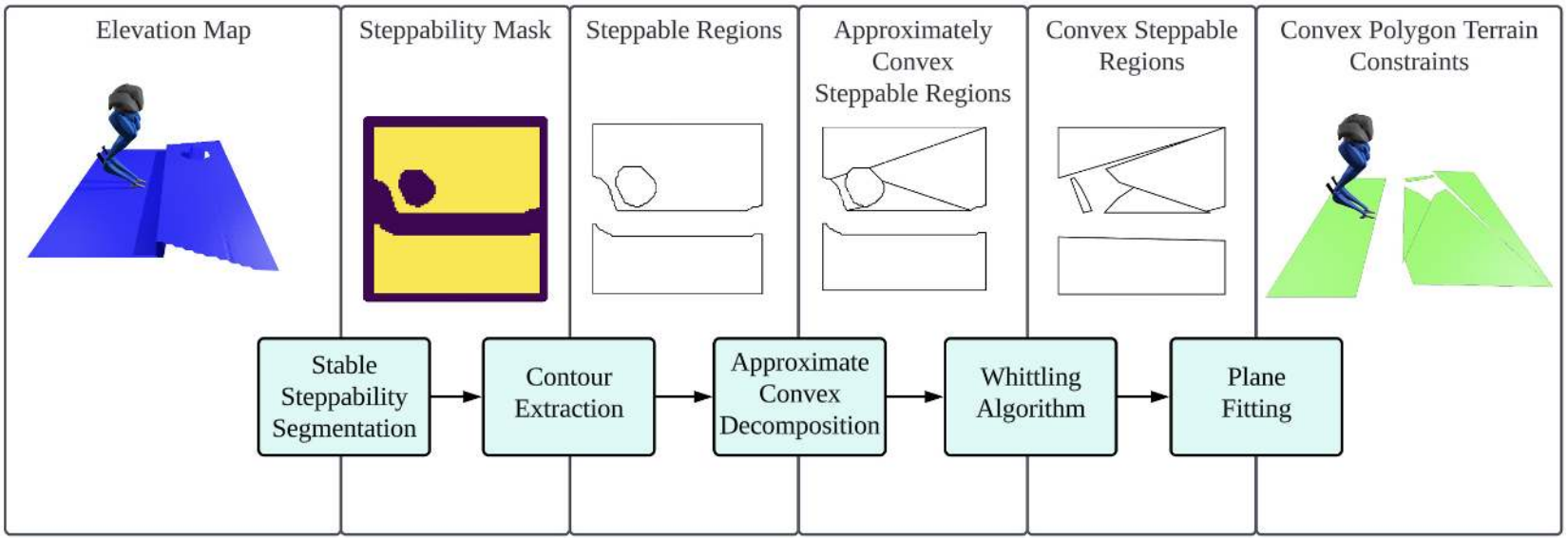


Figure 6.1: Pipeline for converting an elevation map of the terrain into a set of convex polygons which can be used to plan safe footsteps. Our Stable Steppability Segmentation produces a temporally-consistent steppability mask representing a 2D overhead view of the safe terrain. We then extract the boundaries of the safe terrain as non-convex polygons, which we decompose into convex polygons using an algorithm based on approximate convex decomposition.

6.1. Stable Steppability Segmentation

The goal of steppability segmentation is to determine where on the elevation map is safe to step. Because the segmentation determines the foothold constraints for MPFC, it is important that the segmentation algorithm is

- Temporally consistent, despite noise and sensor-fusion artifacts,
- Computed in real time,
- Appropriately conservative.

To accomplish this, we compute various "safety criteria" for whether a pixel is considered safe (Fig. 6.2). A safety criteria is a function transforming the elevation map to a pixel-wise "safety score" in the range $[0, 1]$, where 1 is completely safe, and 0 is unsafe. These safety criteria are fused via their geometric mean to yield an overall safety score. Robustness to noise is accomplished via inpainting and filtering in each safety criteria computation, for example median-filtering the elevation map to remove outliers (Jenelten et al., 2022). Robustness is also achieved by maintaining temporal consistency despite artifacts from state estimate drift and impacts. This temporal consistency is due to the locality of the S3 algorithm, and enhanced by adding hysteresis to the overall safety score based on the previous segmentation. Realtime computation is achieved through the simplicity of our algorithm, and the small size of our elevation map. Because safety criteria are local to each pixel, S3 could also be GPU-parallelized for large elevation maps. The next subsection outlines what we mean by "appropriately conservative" and introduces a curvature-based safety criterion which accomplishes this goal.

6.1.1. Curvature Safety Criterion

During our experiments in (Acosta and Posa, 2023), we used a plane segmentation approach (Miki et al., 2022b), and had difficulty picking a safety margin which avoided tripping over curbs (Fig. 6.3) while not taking excessively large steps over curbs. This experience illustrated the need to step further away from the bottom of a ledge than the top. To penalize terrain which is below edges, we need

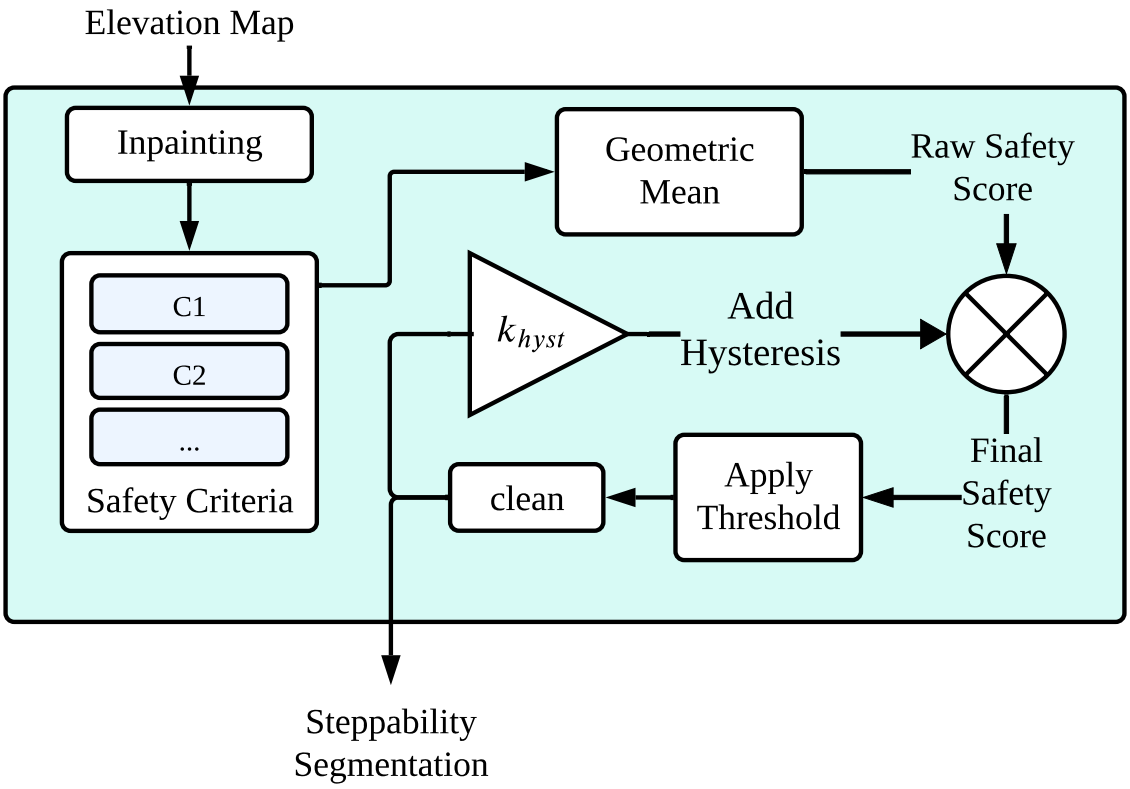


Figure 6.2: Block diagram of S3, our proposed terrain segmentation approach. Safety criteria are combined into an overall safety score for each elevation map pixel, before applying hysteresis to enhance temporal consistency.

to identify terrain that is lower than its surroundings. Treating the elevation map as an image, this looks like applying a kernel that compares the height of each pixel to the average of the pixels around it:

$$\frac{1}{8}\begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{6.1}$$

This particular kernel is a Laplacian kernel, used to compute the curvature of an image, meaning we can use standard image processing tools to efficiently calculate this safety criterion. Letting E be the elevation map, the curvature criterion is computed via Eq. (6.2),

$$c_{curve} = \min(1, \exp(-\alpha_c \text{LoG}(E))) \tag{6.2}$$

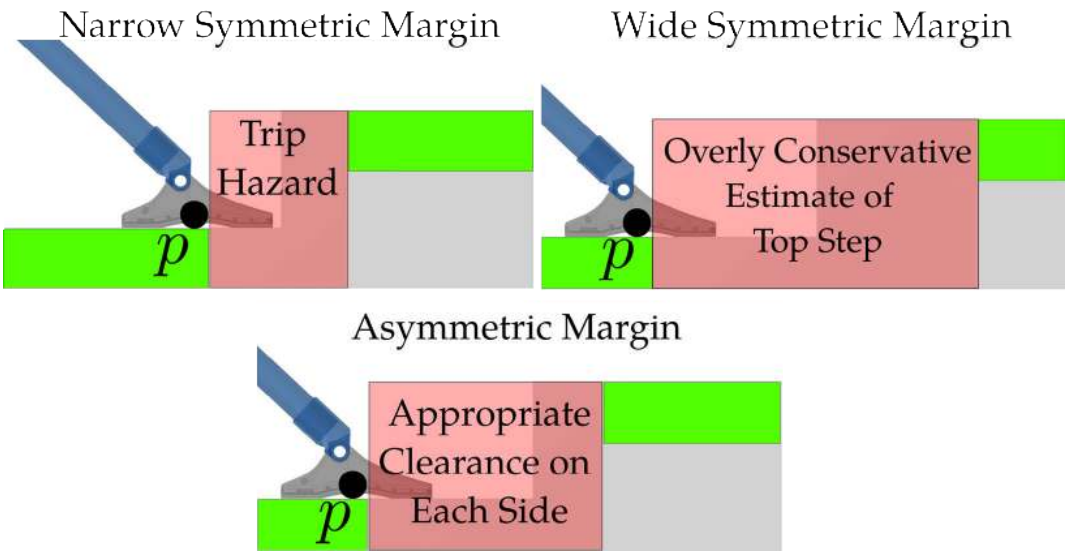


Figure 6.3: Walking over ledges with Cassie requires asymmetric constraints on the footstep position, p, which is mapped to the center of Cassie's foot. The desire to specifically avoid stepping below an edge motivated our curvature based safety criterion, which differentiates between sides of an edge using the sign of the elevation map's Laplacian.

where LoG is the Laplacian of Gaussian filter, which convolves the elevation map first with a Gaussian filter, then takes the Laplacian. The pixel-wise exponential $\exp(-\alpha_c \text{LoG}(E))$ maps regions of positive curvature to the interval $(0, 1]$, with the score exponentially approaching 0 as the curvature increases. The scale factor α_c can be used to tune how aggressively positive curvature is punished. We take the min of the criterion with 1 to ensure that no bonus points are awarded for negative curvature. Penalizing only positive curvature specifically targets area below edges, which poses a tripping hazard. To segment out the edge itself, we introduce an inclination safety criterion, which operates on the estimated normal of the elevation map to penalize steep terrain.

6.1.2. Inclination Safety Criterion

The inclination safety criterion treats steep terrain as unsafe, by considering the magnitude of the z component of the surface normal at each elevation mapping pixel. We estimate the normal using the covariance matrix of the positions around each pixel (Grandia et al., 2022), then square the z component to yield the inclination safety criterion,

$$c_{inc} = n_z(E)^2. \tag{6.3}$$

To give context for how c_{inc} classifies terrain in practice, we pick 0.7 as the final safety threshold for S3. For a pixel which is otherwise considered safe, this means a slope greater than about 33° is unsafe, since $\cos^2(33°) \approx 0.7$.

6.1.3. Combining Safety Criteria

The final safety score is the geometric mean of the score for each criteria, plus a hysteresis value for all pixels classified safe in the previous frame. Pixels with a final score above some threshold are considered safe. The resulting binary image is post-processed to give a 2D view of the safe terrain around the robot. Letting k index the time series of segmentations, the S3 output is given by

$$S_k = \text{clean}\left(\left[\left(\prod_{i=1}^{M} c_i(E_k)\right)^{1/M} + k_{hyst} S_{k-1}\right] > k_{safe}\right)$$

where M is the total number of safety criteria, and $\text{clean}(S) = \text{open}(\text{close}(\text{erode}(S)))$. The erode operation adds a safety margin to account for swing-foot tracking error and the length of Cassie's foot. The open and close operations remove any thin holes or protrusions.

6.2. Convex Planar Decomposition

Finally, we convert the binary steppability mask into a set of convex planar polygons. We identify connected components of steppable terrain from the mask, and extract their outlines as 2D polygons. In general, these are non-convex polygons with holes (caused, for example, by small obstacles or other unsteppable areas), but we require convex foothold constraints for the MPFC. We use a two stage process to find a set of convex polygons whose union is an inner approximation these non-convex polygons. This avoids creating many small triangles like an exact convex decomposition would, leading to fewer mixed integer constraints in the MPFC.

First, we perform approximate convex decomposition (ACD) (Lien and Amato, 2006) on each polygon. ACD returns a decomposition of the original region into polygons which are d-approximately

convex, where d the depth of the largest concave feature.

After filtering out polygons with area less than 0.05 m^2, we find a convex inner-approximation of these nearly convex polygons with a greedy approach we name the whittling algorithm (Algorithm 1), after the way it makes incremental cuts to the polygon. We initialize the output polygon, $\mathcal{P}$ as the convex hull of the original polygon, then take $\mathcal{P}$ to be the intersection of itself with greedily chosen half-spaces until no vertices of the original polygon are contained in the interior $\mathcal{P}$. To reduce the number of cuts we make, we initially sort the vertices by their distance to the boundary of $\mathcal{P}$, handling the innermost vertices first.

Algorithm 1 Whittling Algorithm

Require: Input polygon vertices $V = \{v_0 \dots v_n\}$
  procedure WHITTLE(V)
    $\mathcal{P} \leftarrow$ ConvexHull(V)
    Sort v_i by distance to $\partial\mathcal{P}$
    for all v_i **do**
      if $v_i \in$ Interior($\mathcal{P}$) **then**
        $H =$ MakeCut(v_i, V)
        $\mathcal{P} \leftarrow \mathcal{P} \cap H$
    return $\mathcal{P}$

MakeCut(V, v_i) is a nonlinear program inspired by maximum margin classification (Boser et al., 1992) which finds a such that the half-space $H = \{x \mid a^T(x - v_i) \leq 0\}$ contains as much of V as possible:

$$a = \arg\min_a \sum_{j \neq i} \max(a^T(v_i - v_j), 0)^2$$
$$\text{subject to } \|a\|_2^2 = 1. \tag{6.4}$$

We solve (6.4) using a custom gradient-based solver, which we detail in Section 6.2.1. Using the normal of the closest face of $\mathcal{P}$ to v_i provides a high-quality initial guess for the solver.

To fit these polygons to the terrain, we project the 2D vertices onto the elevation map to recover the 3D position of each vertex. We then use least-squares to find the best fit plane to these vertices,

yielding our final polygon representation.

6.2.1. Whittling Algorithm Cut Solver

We briefly present a solver for optimization over S^1, which we use to solve (6.4) quickly online. Given an optimization problem

$$\underset{x \in \mathbb{R}^2}{\text{minimize}} \; f(x) \tag{6.5}$$

$$\text{subject to } \|x\|_2^2 = 1, \tag{6.6}$$

the associated first-order optimality conditions are

$$\|x\|_2^2 = 1 \tag{6.7}$$

$$\nabla f(x) + \nu x = 0 \tag{6.8}$$

where $\nu \in \mathbb{R}$ is a Lagrange multiplier for the unit-norm constraint. The main idea of our solver is that because we are optimizing over the unit circle, we rotate x in the direction which decreases the cost until $\nabla f(x)$ is parallel to x, which satisfies the optimality conditions. Our solver is summarized in Algorithm 2.

Algorithm 2 MakeCut Solver

Require: Cost function f, Initial guess $x \in S^1$, Optimality Tolerance ϵ, Line search parameters $\alpha > 0, \beta \in (0, 1)$

```
procedure SOLVE(f, x, ϵ)
    θ ← ∞
    while |θ| > ϵ do
        θ ← (∇f(x) − x⟨∇f(x), x⟩) × x
        t ← α/|θ|
        while f(Rotate(θt, x)) > f(x) do t ← βt
        x ← Rotate(θt, x)
    return x
```

In Algorithm 2, the Rotate subroutine is defined via

$$\text{Rotate}(\theta, x) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} x.$$

The θ update finds the direction to rotate x by considering the component of ∇f orthogonal to x, using the cross product with x to convert this direction into a scalar rotation angle. We then perform a line search, starting with a fixed initial step size α for improved convergence speed. As an implementation note, we re-normalize x at each iteration to avoid drift in the unit-norm constraint.

6.3. Results

We present results to support our claims of S3's improved temporal consistency and faster run time compared to explicit plane segmentation. We use elevation mapping data from three terrains to evaluate segmentation performance (Fig. 6.4). Each dataset contains a time series of elevation maps collected during perceptive locomotion experiments on Cassie using MPFC. The full system is described in Chapter 7. For our current purposes it is sufficient to consider just the time series of elevation maps, abstracted from how these timeseries were collected. The **Lab** terrain establishes a baseline for each method in an ideal environment, where state estimate z-drift is the main challenge. The **Brick Steps** terrain features a set of brick steps where the bricks have settled over time, making the steps uneven, and unlikely to be segmented into a single plane by plane-segmentation methods. Similarly, the **Grass** terrain is challenging for plane segmentation approaches because our stance foot drift-correction conflicts with the height of the point cloud, introducing artifacts into the elevation map.

We use the plane segmentation module developed by Miki at al. in the `elevation_mapping_cupy` software package Miki et al. (2022b) with default parameters (hereafter labeled EM_cupy) as a plane segmentation baseline. This algorithm has a similar structure to S3, starting by filtering the elevation map, classifying each cell as steppable or not, and then (unlike S3), trying to segment the steppable cells into planes. Each connected component of steppable terrain is checked for planarity, and if it fails, RANSAC Schnabel et al. (2007) is used to find smaller planes within that con-

Figure 6.4: The environments used to collect data for benchmarking the perception stack's performance. From top to bottom the terrains are **Lab**, **Brick Steps**, and **Grass**

nected component. The authors of Miki et al. (2022b) also provide the option to disable RANSAC plane refinement, instead accepting or rejecting each connected component of steppable terrain in its entirety based on the estimated surface normal. We also test this variant, henceforth labeled EM_cupy_NR, where NR denotes "No RANSAC" or "No Refinement." Because EM_cupy_NR is identical to the default EM_cupy algorithm except for lacking a global planarity requirement on steppable regions, these results will support our argument that explicit plane segmentation is particularly brittle. Table 6.1 summarizes the differences between S3 and the baselines.

6.3.1. Computation Time

As a benchmark against other segmentation approaches, we compare the run times of S3, EM_cupy, and EM_cupy_NR for each test environment in Fig. 6.5, and find S3 to be the most consistent, with the lowest worst-case computation time.

6.3.2. S3 Temporal Consistency

We measure the temporal consistency of each segmentation approach via the intersection over union (IoU) of consecutive segmentations. IoU measures the ratio of pixels labeled as safe in both segmentation frames to the number of pixels labeled safe in either frame. Because data is lost when the elevation map moves relative to the world, we restrict the IoU computation to pixels which are present in both frames. A frame-to-frame IoU of 1 represents perfect temporal consistency, and 0 represents no overlapping safe terrain between segmentations.

The distributions of frame-to-frame IoU for one minute of walking data in each environment are shown in Fig. 6.5. Our approach consistently achieves an IoU close to 1 across environments, representing excellent temporal consistency, while EM_cupy has notably lower IoU even in the lab setting, due to imperfect depth estimation and artifacts from sensor fusion.

Table 6.1: Comparison of S3 and Plane Segmentation Baselines As Benchmarked

	S3 (Ours)	**EM_cupy**	**EM_cupy_NR**
Steppability Criteria	Curvature, Inclination	Roughness, Inclination	Roughness, Inclination
Incorporates History	Yes	No	No
Plane Refinement	None	RANSAC Schnabel et al. (2007)	Reject regions with slope $\geq 30°$
Inpainting Method	Navier-Stokes Bertalmio et al. (2001)	Least Neighboring Value	Least Neighboring Value

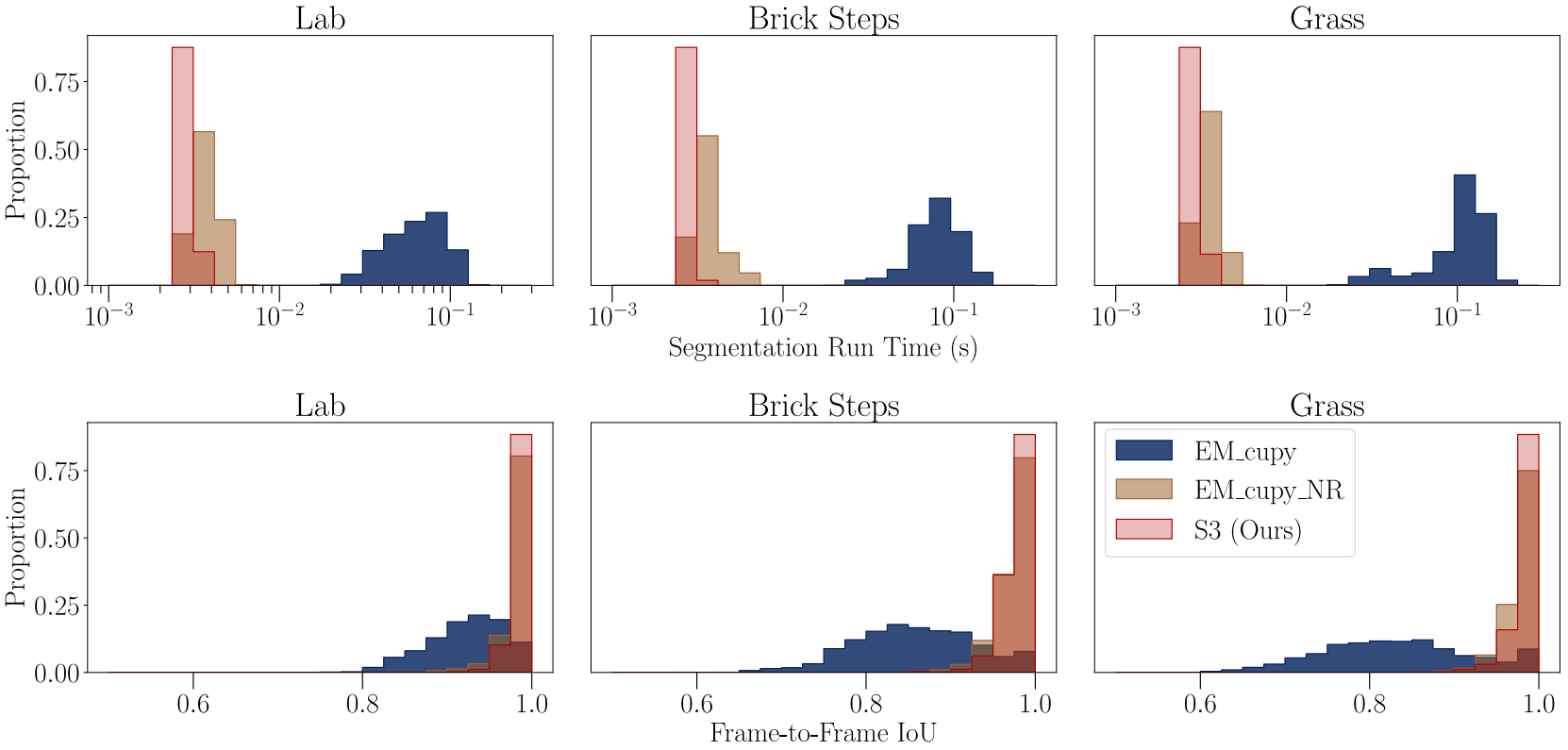


Figure 6.5: Offline benchmark of S3 compared to plane segmentation baselines. **Top:** Histogram of the run time of each segmentation algorithm over 60 seconds of elevation mapping data from each test environment. Benchmark was run on an Apple Macbook Pro with an M1 Max CPU, 10 cores, and 64 GB of RAM. S3 has the fastest and most consistent run times. EM_cupy is the slowest, with a highly variable run time, due to the repeated use of RANSAC to refine the plane segmentation. **Bottom:** Histogram of the frame-to-frame IoU of the safe terrain segmentation over the same datasets. Our method reliably achieves a frame-to-frame IoU close to 1 across environments, representing excellent temporal consistency.

EM_cupy_NR has similar temporal consistency to S3, though the lack of hysteresis contributes to small holes which appear and disappear. The improved temporal consistency of EM_cupy_NR over the default EM_cupy shows that the plane-segmentation step in particular is brittle, rather than other design choices like inpainting or steppability criteria This highlights that requiring regions to be globally planar is mostly responsible for the poor temporal consistency of EM_cupy. The segmentation output from each algorithm at 1 second intervals is shown in Fig. 6.6, and animations of the segmentation state are shown in the supplemental video.

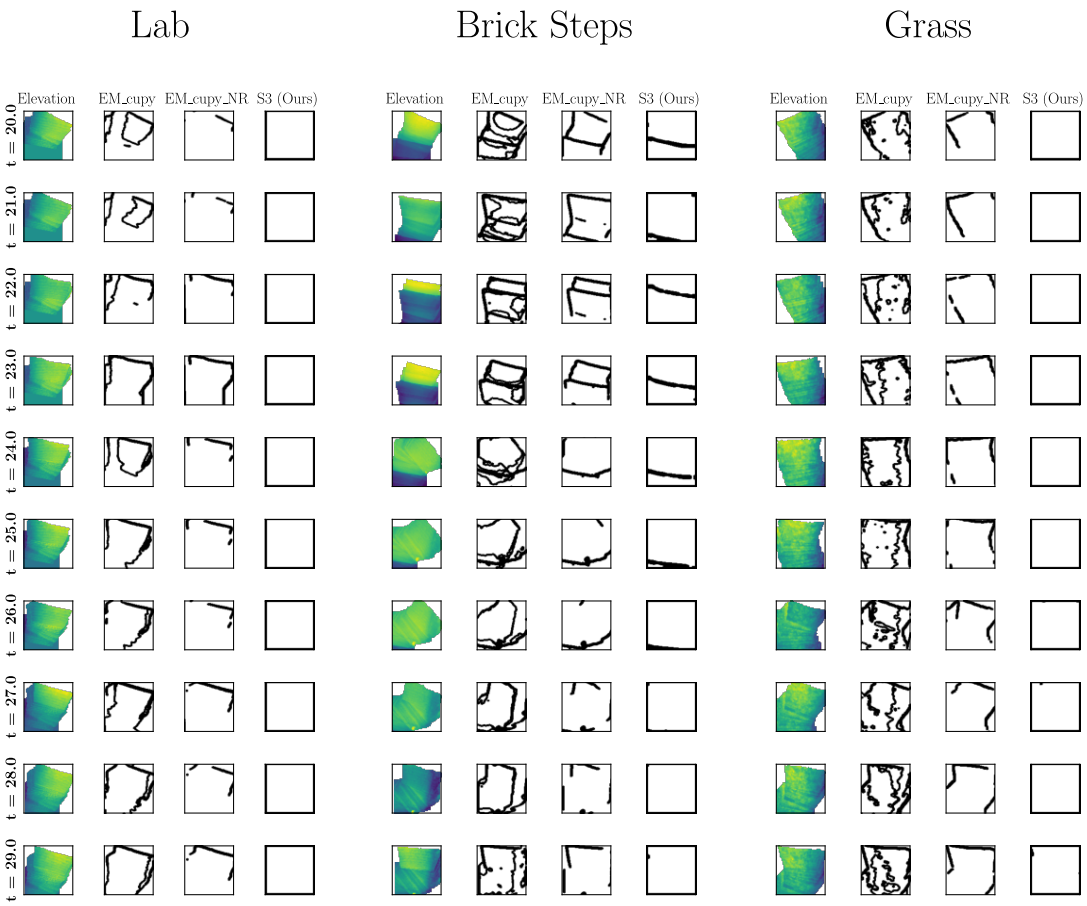


Figure 6.6: Tiles showing the output of each segmentation method for each evaluation environment at 1 second intervals. In the **Lab** and **Grass** environments, the use of Navier-Stokes based inpainting allows S3 to correctly identify the entire elevation map as steppable. The S3 segmentation experiences minimal "flickering" of the steppable terrain compared to the baselines. Animations of these segmentation results can be seen in the supplemental video.

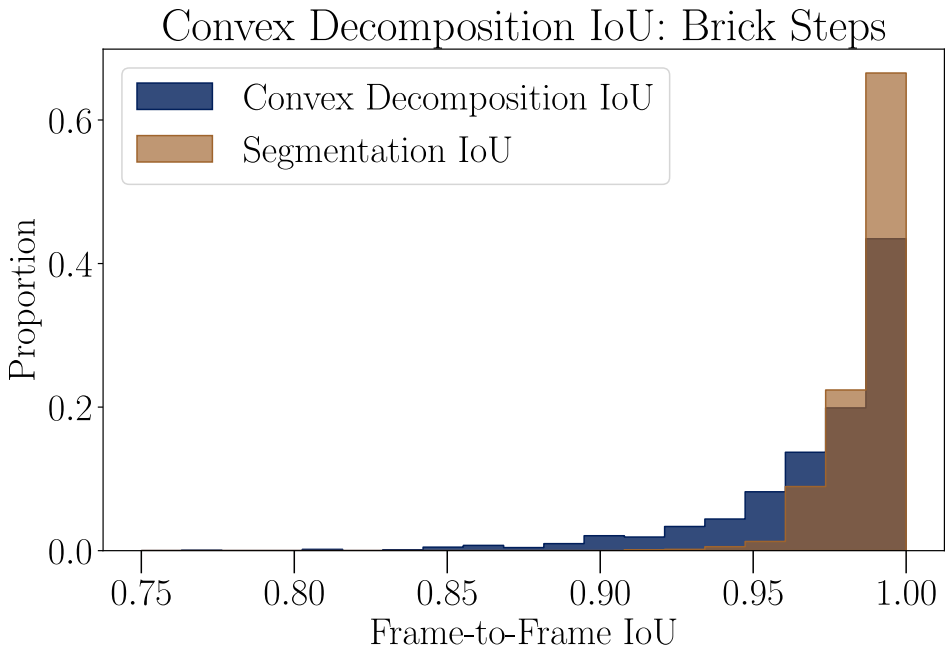


Figure 6.7: The final convex decomposition has a similarly shaped IoU distribution to the safe terrain segmentation, though is less consistent overall.

6.3.3. Convex Polygon Temporal Consistency

This section verifies that a temporally consistent terrain segmentation ultimately leads to a temporally consistent convex polygon decomposition. Because MPFC is free to pick any foothold for each solve, we compute the frame-to-frame IoU of the terrain covered by each convex decomposition, rather than the consistency of individual polygons. We compute the IoU by sampling. Each elevation map cell is marked as safe if its 2D position is covered by a convex polygon. We then compute the IoU of the safe cells corresponding to each consecutive convex decomposition. The distribution of IoU for the safe terrain vs. for the convex decomposition is shown in Fig. 6.7.

6.3.4. S3 Hysteresis and Moving Obstacles

For our hardware experiments, we chose a hysteresis value of 0.6, with a safety threshold of 0.7. This high level of hysteresis enhanced the consistency of the segmentation for the terrains we tested on, however less hysteresis may be desirable in dynamic environments. To determine the appropriate hysteresis for different scenarios, we provide two analyses. First, we perform experiments with moving obstacles, by tossing 15 cm foam cubes into the scene and counting how many are segmented

out for each hysteresis condition. An obstacle is defined as segmented out if it creates a hole in the terrain segmentation before it comes to rest. These results are shown in Table 6.2 and in the supplemental video. Second, we show the distribution of frame-to-frame IoU values for varying levels of hysteresis on the **Brick Steps** in Fig. 6.8. The lowest observed IoU was 0.78, even without hysteresis, and a hysteresis factor as low as 0.3 performed similarly to the chosen value of 0.6. These results suggest that a hysteresis factor between 0.3 and 0.4 can enhance the stability of the terrain segmentation while maintaining correctness in dynamic environments[4].

Table 6.2: Moving Obstacles Segmented by S3 vs. Hysteresis parameter ↑. An obstacle is defined as segmented out if it creates a hole in the terrain segmentation before it comes to rest.

k_{hyst}	0.0	0.1	0.2	0.3	0.4	0.5	0.6
Standing	3/3	3/3	3/3	3/3	3/3	2/3	0/3
Walking	3/3	3/3	3/3	3/3	3/3	2/3	0/3

6.4. Discussion

This section discusses implementation details and design choices introduced to handle edge cases and increase the robustness of our S3 implementation. We then cover limitations of the S3 framework and opportunities for future work, again deferring the systems integration aspect between S3 and the locomotion controller to Chapter 7.

One systems-level limitation is that we use single threaded CPU implementations of elevation mapping and S3, limiting the map size and resolution which can be handled in real time. Existing GPU-based elevation mapping implementations (Miki et al., 2022b) could be used with a GPU implementation of S3 to handle larger or more detailed maps.

Additionally, this chapter introduces heuristic steppability criteria, which demonstrate the temporal consistency and computational efficiency advantages of S3 compared to plane segmentation, but do not investigate other potential benefits of S3 as a general framework. For example, one failure case on hardware involved dried leaves which had accumulated underneath the edge of a step, filling in the space and resulting in the edge being classified as steppable. This could be avoided by

[4]Regrettably, our outdoor experiments do not include results with a hysteresis other than 0.6, as malfunctions of the physical Cassie robot prevented us from performing additional hardware experiments after we completed the moving-object experiments.

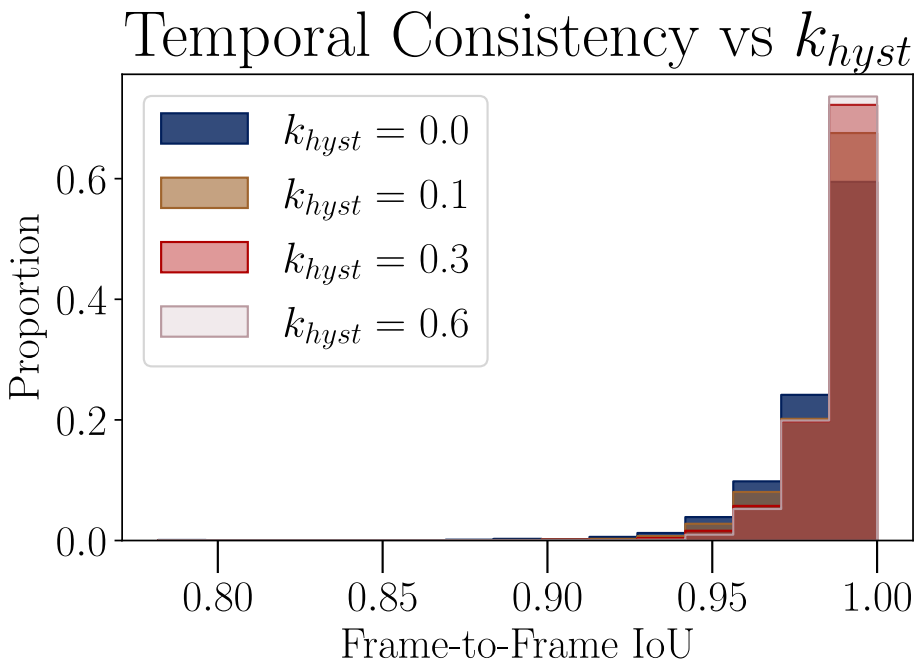


Figure 6.8: Frame-to-Frame IoU of the S3 terrain segmentation results with varying levels of hysteresis, evaluated on the **Brick Steps** data. The lowest observed IoU was 0.78, even without hysteresis, in contrast to both of the plane segmentation baselines, whose the IoU varied across the entire [0, 1] interval.

incorporating a higher level semantic segmentation as an additional safety criterion.

6.5. Conclusion

This chapter introduces Stable Steppability Segmentation (S3) for classifying safe terrain, and designs a broader pipeline around S3 for converting an elevation map into a convex polygon terrain decomposition. S3 achieves higher temporal consistency and faster run times than previous approaches by omitting an explicit plane-segmentation step. The segmentation is instead performed by considering safety criteria which are local to each elevation mapping pixel. The locality of the safety criteria allows the temporal consistency of the algorithm to be further enhanced easily by adding hysteresis to the plane segmentation. We validate these claims of improved run time and temporal consistency on data from bipedal walking over several real world terrain types, including terrains which are antagonistic toward plane-segmentation. We introduce a convex decomposition

procedure for extracting convex planar polygons from the steppability segmentation and elevation map using approximate convex decomposition and a greedy algorithm for finding large inscribed convex polygons. The next chapter validates the full perceptive walking framework through hardware experiments on the Cassie biped. In contrast to our original implementation using plane segmentation (Acosta and Posa, 2023), S3 allows the robot to walk continuously with perception in the loop due to its temporal consistency, despite artifacts from state-estimate drift and impacts.

CHAPTER 7

Full-Stack Perceptive Locomotion Experiments

Parts of this chapter were previously published as parts of Brian Acosta and Michael Posa. Perceptive Mixed-Integer Footstep Control for Underactuated Bipedal Walking on Rough Terrain. *arXiv preprint arXiv:2501.19391*, January 2025.

Supplemental video for Chapters 5, 6, and 7: https://youtu.be/JK16KJXJxi4

This chapter synthesizes the contributions of Chapter 5 and Chapter 6 into a full stack perceptive locomotion architecture for the bipedal robot Cassie (Fig. 7.1). This architecture enables Cassie to walk over previously unseen terrain by identifying safe terrain and planning stabilizing footsteps subject to non-convex terrain constraints in real time. Because key results for the individual components have been presented in Chapter 5 and Chapter 6, this chapter will focus on documenting the system implementation used to produce these results, and discussing aspects of the overall system performance related to interactions between the perception system and the locomotion controller. The key consideration for how the controller and perception system interact is that restricting steppable area restricts the control authority of the ALIP model, especially in the lateral direction, which lacks ankle torque. This increases the need for a consistent terrain segmentation and means that a more conservative segmentation may not always be safer when it comes to closed-loop locomotion performance.

7.1. Experimental Setup

This section explains the practical implementation of MPFC and our perception stack as an integrated system. The parameters used for S3, MPFC, and their supporting algorithms are given in Appendix B. The full perception and control system consists of six processes across three separate computers (Fig. 7.2). Cassie's target PC runs a Simulink Real-Time application which publishes joint positions and velocities and IMU data, at 2kHz, and subscribes to torque commands. Communication between the target PC and Cassie's onboard Intel NUC occurs over UDP. The NUC

runs the state estimator and a torque publisher to communicate with the target PC, and the OSC process, which includes the CoM and swing foot planner.

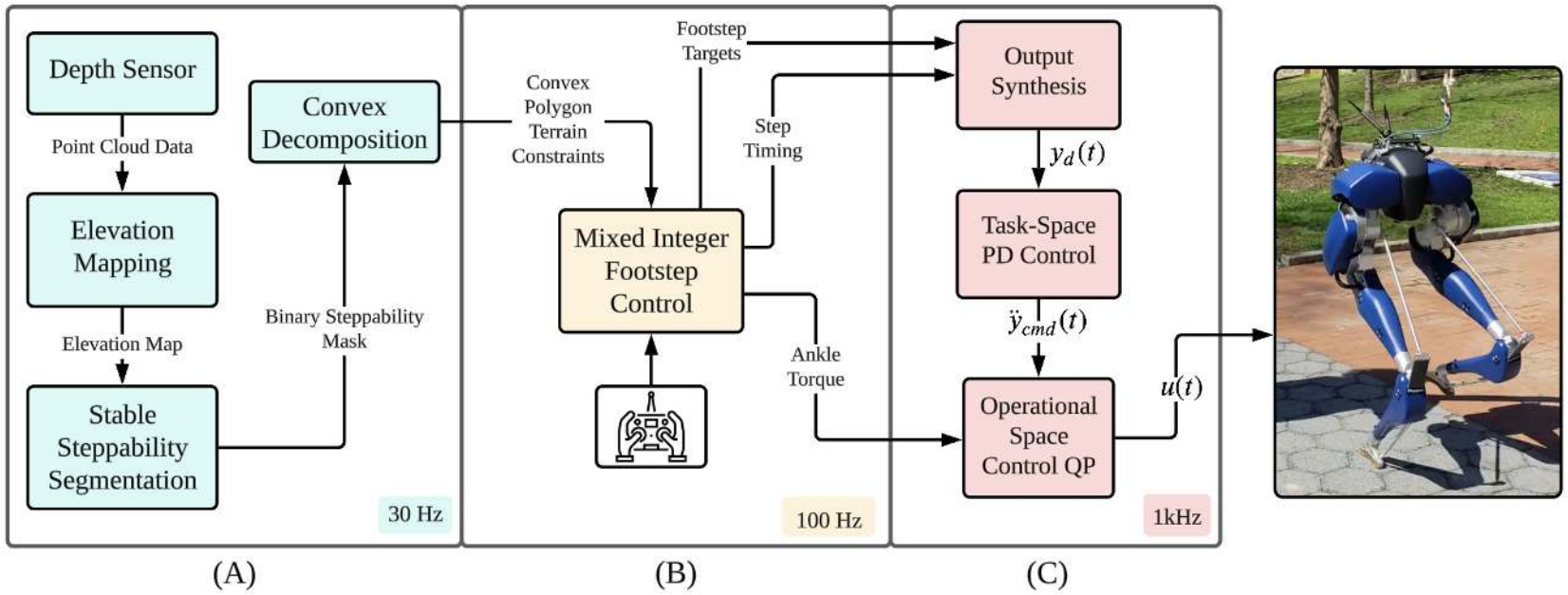


Figure 7.1: The perception and control stack proposed in this chapter to achieve underactuated walking over discontinuous terrain. Our perception stack (A) generates convex polygon foothold constraints for MPFC, a mixed-integer MPC style footstep planner (B). MPFC sends the next footstep, step timing adaptation, and ankle torque plan to a low-level operational-space-control process (C) which performs kHz level torque control.

The perception stack and MPFC are run on an off-board ThinkPad p15 Laptop with an 8-core, 2.3 GHz Intel 1180H processor and 24 GB of RAM. The perception stack performs elevation mapping, terrain segmentation, and convex decomposition in one thread, and has a second thread to poll the Intel RealSense. The state estimator, operational space controller, torque publisher, perception stack, and MPFC communicate over LCM (Huang et al., 2010) for low latency.

Except for the low-level target PC, all processes use the Drake systems framework to drive their operation (Russ Tedrake and the Drake Development Team, 2019). We solve the MPFC problem using Gurobi, and the OSC QP using FCCQP (Acosta, 2024). We use the contact-aided invariant extended Kalman filter developed by Hartley et al. (Hartley et al., 2020) to estimate the pose and velocity of the floating base.

Open source code for all of our contributed components will be provided in `dairlib`[5].

Figure 7.2: The experiment setup for the Physical Cassie robot showing which computers run which processes. The robot is attached to a safety tether which remains slack while the robot is walking.

7.1.1. RealSense D455 Depth Camera

The RealSense is mounted to Cassie's pelvis, pointed downward toward the terrain in front of the robot. We use `librealsense2` to subscribe to RealSense frames via a dedicated polling thread, with the perception stack thread accessing these frames through a shared buffer. We apply a decimation

[5]https://github.com/DAIRLab/dairlib

filter to reduce point cloud density.

7.1.2. Robot-Centric Elevation Mapping

We use the framework of Fankhauser et al. (2018) to construct a robot-centric elevation map of the terrain. This framework represents the terrain as a regular grid, with the height of each cell updated by point cloud measurements through a Kalman filter. Because Cassie's legs are visible in the camera frame, we crop out any points inside bounding boxes around Cassie's leg links. State estimate z-drift is a well-known source of elevation mapping artifacts which must be corrected for an accurate terrain estimate. Related works use perception information to correct drift (Miki et al., 2022b; Bin et al., 2024), however this correction is not always sufficient when the walking motion generates non-negligible impacts (Grandia et al., 2022), as is the case for most Cassie walking controllers. To strongly correct for state estimate z-drift, before each point cloud update, we adjust the height of the elevation map by adding the height difference between the elevation map and the current stance foot. To account for outliers, we calculate the elevation map height as the median of a 4x4 pixel grid, centered at the contact point.

7.2. Results

The perception and control architecture presented in this thesis enables Cassie to walk over previously unseen terrain by identifying safe terrain and planning stabilizing footsteps subject to non-convex terrain constraints in real time. This section presents experiments to show these capabilities and support our key claims. We perform simulation experiments to quantify the performance gap between walking over known vs. online-identified safe terrain. On hardware, we showcase underactuated walking over discontinuous terrain with Cassie, summarize the capabilities of MPFC and S3 as a complete system and highlighting the performance improvements as a result of the contributions in Chapters 5 and 6.

7.2.1. Simulation Experiments

Our perceptive locomotion simulation is identical to the simulation developed for Chapter 5, except that it simulates the invariant EKF Hartley et al. (2020) to provide state estimates to OSC, MPFC, and the elevation mapping system, and uses a simulated depth sensor as input to the perception

pipeline. The S3 steppable area is more conservative than ground truth, resulting in a traversable beam width of 35 cm and stair depth of 40 cm with perception, compared to 23 cm and 27 cm respectively for ground truth terrain.

Repeating the stepping stone experiment from Section 5.3.1 shows that the more conservative steppable terrain estimation is particularly consequential for smaller footholds (Fig. 7.3).

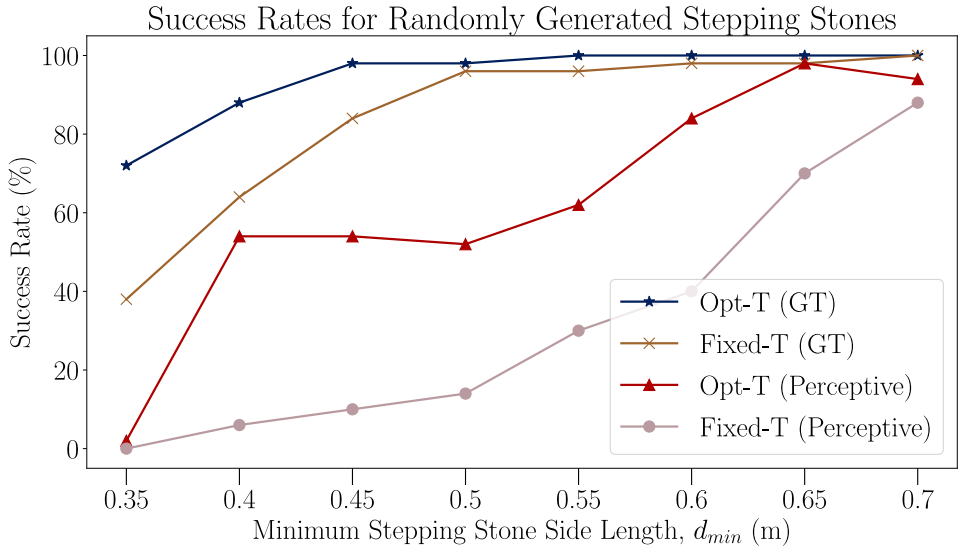


Figure 7.3: Success rates for walking across randomly-generated stepping stones in simulation, where the stepping stone sizes are distributed according to Table 5.1 based on the minimum side lenght, d_{min}. Results with step-timing optimization are labeled Opt-T, and results without step timing optimization are labeled Fixed-T. We report results with ground truth state and terrain (GT), as well as with perceptive terrain using S3 (Perceptive). The lower success rate using perception is primarily due to isotropic safety margin in S3 reducing the lateral steppable area compared to ground-truth, which accounts for the width and length of the foot separately.

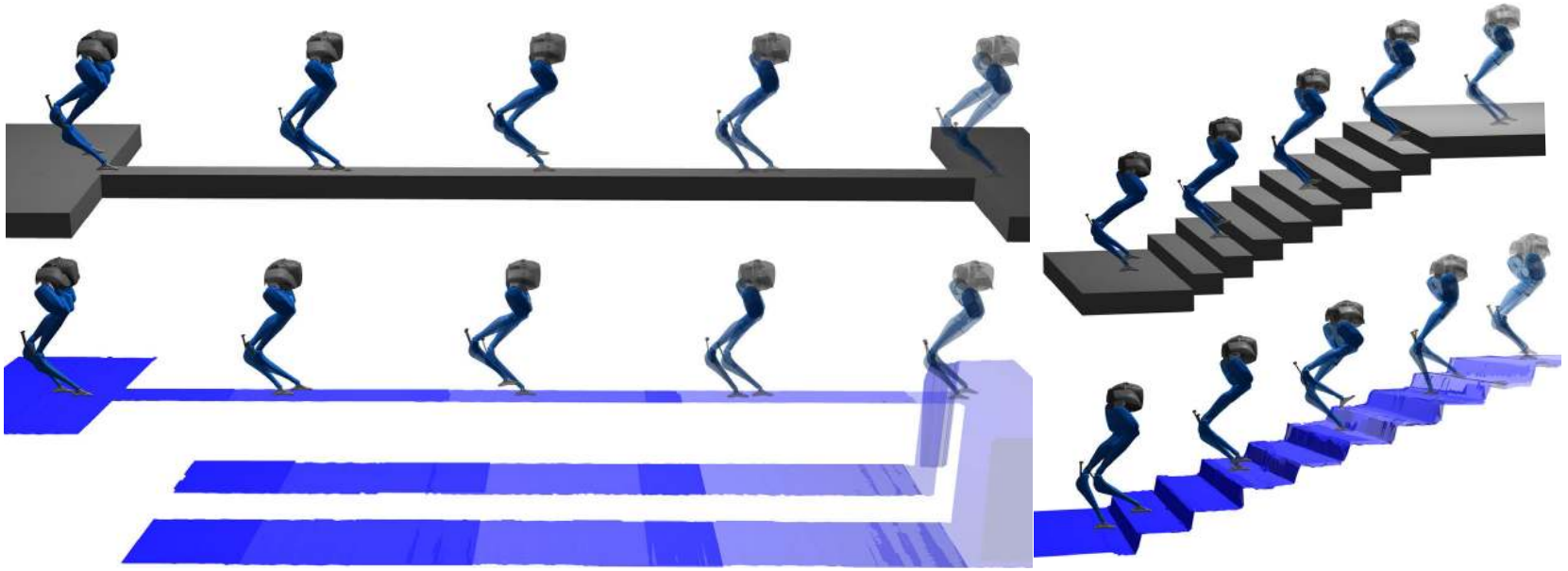


Figure 7.4: MPFC simulation experiments using ground truth vs. perceptive terrain. **Top:** we use ground truth terrain information to walk over a 23 cm wide beam, and stairs with a rise of 15 cm and a depth of 27 cm. **Bottom:** Displaying the elevation map for walking over the same terrain types using S3. Safety margin in S3 results in less steppable area than for ground truth, so the minimum traversable dimensions are increased to 35 cm wide for the beam and 40 cm deep for the stairs.

7.2.2. Hardware Experiments

We demonstrate Cassie traversing discontinuous terrains using the presented perception and control stack. Given the feature size limitations which can be effectively segmented by S3, we primarily test on the **Brick Steps** terrain from the previous chapter, which we show another view of in Fig. 7.6, as well as a similar set of steps nearby (Fig. 7.5).

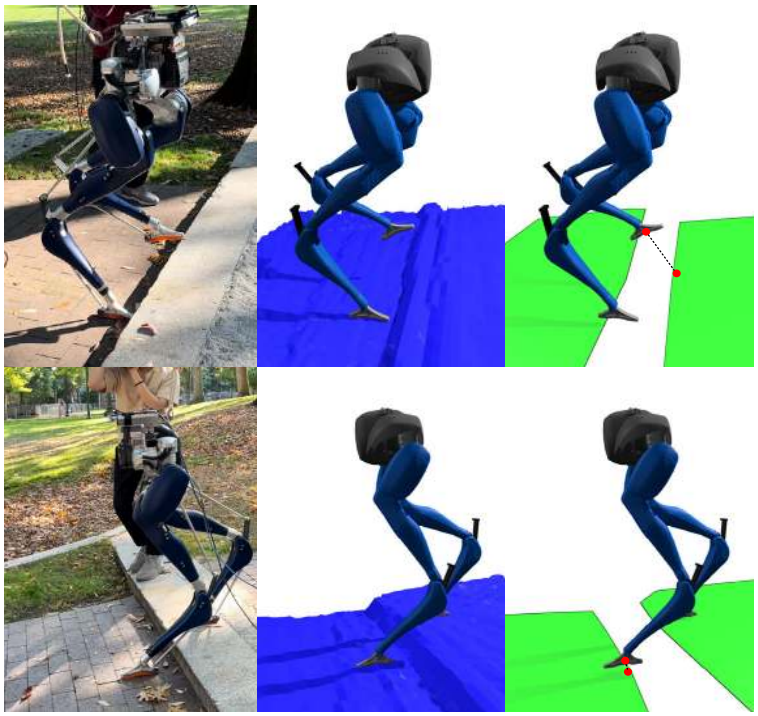


Figure 7.5: Cassie walks up and down brick steps using the perception and control framework developed in this thesis. Left: the physical robot and steps. Middle: an elevation map of the steps. Right: the convex decomposition of the safe terrain.

The fastest time to ascend the steps in Fig. 7.6 was 13 seconds, from crossing edge of the bottom step to the entire robot being above the top step. However, slower walking speeds were more reliable given the limited friction of the steps. The longest continuous trial was 4:40 minutes, wherein Cassie ascended the steps four times and descended them three times. In this trial, Cassie fell stepping down the top step, which has the largest height change of all of the steps at 16 cm (Fig. 7.7). Since this was the largest height-change we traversed on hardware, and the step which was least reliably traversable, especially descending, we will dedicate some space here to discussing the challenges we

Figure 7.6: Another view of Cassie descending a set of steps using our full-stack perceptive locomotion architecture.

observe crossing this step. First, significant CoM and swing foot accelerations are needed. The swing foot acceleration sometimes compressed Cassie's leaf springs enough to trigger the state estimator contact detection, leading to incorrect velocity and pose estimates of the robot. Because the OSC has only a one-step preview of the footstep plan, the CoM must complete the entire height change in a single stride, creating large ground reaction forces, which was ultimately less forgiving if the robot slipped. Finally, when stepping down, the swing-foot could create a large impact on touchdown, leading to torque spikes and jumps in the angular momentum estimate.

In addition to the brick steps, we show Cassie stepping up over a more un-structured curb onto a grassy knoll in Fig. 7.8. Further trials, including trials where Cassie deviates from the "obvious" foothold sequence in order to take recovery steps, can be seen in the supplemental video [6].

Summary

Compared to our deadbeat ALIP footstep planner based on (Gong and Grizzle, 2021), MPFC is more robust, even on hard, flat surfaces, due to the inclusion of workspace constraints, ankle torque, and step timing optimization. MPFC and S3 also enable Cassie to walk up and down steps and curbs

[6] https://youtu.be/uhgyEdGdtVc?si=9IujrqUl_2kiF6dW

Figure 7.7: The top of the brick steps has the largest height change at 16 cm, which sometimes challenges the OSC's ability to track the CoM and swing foot, and creates large impacts when stepping down.

up to 16 cm tall when each step is deep enough to have a valid S3 segmentation. In contrast to our original perception implementation in (Acosta and Posa, 2023), using S3 for terrain segmentation allows the robot to walk continuously with perception in the loop due to its temporal consistency, despite artifacts from state-estimate drift and impacts. This is the case even on grass, where proprioceptive and exteroceptive ground height estimates conflict. We discuss limitations of our full stack implementation in Section 7.3.1.

Figure 7.8: Cassie steps over a high curb onto a grassy knoll.

7.3. Discussion

This section discusses implementation details, limitations, and failure modes. First, we discuss design choices introduced to handle edge cases and increase the robustness of our implementation.

Inpainting

Because we only use a single depth camera, whose field of view does not span the entire diagonal of the elevation map, we lack elevation data for terrain near the robot when not walking straight forward, leaving the question of how S3 should classify these cells. During our hardware testing, classifying these cells as unsafe caused the robot to fall when the operator drove the robot toward unmapped regions. We solve this by inpainting the missing portions of the elevation map using the Navier-Stokes based method implemented in OpenCV Bertalmio et al. (2001), before inputting the elevation map to S3. This method matches the value and gradient of the image at the boundary of the missing terrain. For many real world terrains, this continuous extrapolation is a safe assumption, since obstacles extend uni-directionally across the entire map. In more dangerous environments, and when missing elevation map values are primarily due to occlusions rather than a lack of sensor coverage, such as in our simulation stepping stone experiments, a more conservative inpainting scheme such as least-neighboring-value (Miki et al., 2022b) is appropriate. The ideal solution would include additional depth sensors, however our solution highlights a general theme: due to Cassie's underactuation, it is often safer to resolve *ambiguous* design decisions by favoring steppability.

ALIP State Estimation

Impacts during touchdown and compliance in Cassie's hip-roll joints can cause undesirable spikes and oscillations in the lateral floating base velocity estimate, and therefore the angular momentum estimate. We increase our controller's robustness to these issues by using a Kalman filter with ALIP dynamics to smooth our estimate of the ALIP state during single support. We use (4.11) for the dynamics model, with full-state measurement, and assume a much higher measurement noise for the angular momentum than for the CoM position.

Figure 7.9: Our proposed system naturally handles terrain with no obvious planar approximation. Despite constantly varying height and surface normals, S3 classifies the entirety of this sinusoidal terrain as safe, resulting in a single steppable region matching the extents of the map. Because ALIP dynamics are height-independent, the z-coordinate of each planned footstep does not affect the rest of the MPC solution. Therefore, we use the elevation map to determine the footstep height sent to the low-level OSC.

Footstep Height Lookup

Before sending a footstep command to the OSC, we refine the vertical footstep position by looking up the height of the planned footstep position on a smoothed, inpainted copy of the elevation map. Because the ALIP dynamics do not depend on the vertical footstep position, we do not need to propagate this adjustment back to MPFC. In addition to increasing the practical robustness of the system, this allows Cassie to walk on undulating terrain without any modifications to the perception or control stack (Fig. 7.9).

7.3.1. Limitations, Failure Modes, and Future Work

At a systems level, the fast swing foot motions and CoM height changes required to walk on steps pushed the boundaries of what could be tracked with our OSC, limiting the step heights traversable on hardware to 16 cm. More challenging terrains will require considering swing-foot and vertical CoM dynamics at the MPC level, either by incorporating more detailed dynamics into MPFC, or by using whole-body MPC to realize the MPFC footstep plans.

The limitation most most insufficiently addressed by this work was slipping. The robot would slip as a result of other more minor limitations, or simply because of the low friction of the tested environments. Most often, the robot would fall immediately upon slipping, but if it recovered, the slip could introduce large errors into the elevation map which lead failure from an incorrect segmentation or error in the estimated ground height. The likelihood of slips could be reduced by

more conservatively constraining the workspace of the ALIP model (effectively the friction cone), but this cannot entirely eliminate the possibility. This vulnerability highlights that like all model-based approaches, ours can be brittle to gaps between modeling assumptions and the real world. On the perception side, S3 prevents inconsistent segmentation of reasonable elevation maps from causing failures, but cannot correct maps which inaccurately reflect the real environment. For situations such as tall grass, or recovering from slip-induced errors, methods are needed which can adaptively rely on either perception or proprioception, and recognize and reset invalid maps.

7.4. Conclusion

This chapter presented the experimental setup and systems integration considerations for full-stack perceptive locomotion on Cassie, leveraging the contributions from Chapter 5 and Chapter 6 of this thesis. To our knowledge, this is the first hardware demonstration of a line-foot biped using vision and model-based control to navigate discontinuous terrain. Our hardware experiments highlight the need for balance between appropriately conservative terrain segmentation and preserving control authority via foot placement for underactuated bipeds. Future work remains to expand the framework to handle more challenging terrains, especially in terms of robustness to slip and blending proprioceptive and perceptive terrain estimation.

CHAPTER 8

Cascaded Fidelity MPFC

This chapter aims to resolve two limitations of MPFC. First, the exponential complexity of MPFC in the planning horizon prevents real-time planning further than two steps ahead, reducing the closed-loop stability of MPFC on more challenging terrains. This exponential complexity is driven by coupling of the optimal foothold choice across the planning horizon. We propose a new problem formulation and general purpose solver based on the Alternating Direction Method of Multipliers (ADMM) which handles the foothold constraints individually for each footstep, removing the combinatorial complexity of MPFC at the cost of optimality and feasibility guarantees. We show via an ablation study over stepping stones that for a sufficiently long planning horizon, this strategy is more successful at traversing challenging terrains than the MIQP solver with a two-step horizon.

Second, the step-to-step ALIP model does not consider swing-foot or vertical CoM dynamics, making it difficult to execute the MPFC plan over large height changes. This could be resolved by considering the full-order robot dynamics in the footstep planning problem. Whole-body MPC can control highly dynamic locomotion over pre-defined stepping stones (Grandia et al., 2022), and problem formulations have been proposed which adapt footstep plans to stepping stone constraints within the MPC problem (Shim et al., 2023). However, applications of this second controller to bipedal locomotion have been restricted to relatively simple terrains. In order to walk over more complex terrains, a longer planning horizon than is currently achievable for whole-body MPC is likely necessary. To that end, we propose using a cascaded-fidelity MPFC formulation (CF-MPFC), and leveraging our new ADMM solver for efficient sequential quadratic programming with stepping stone constraints. The cascaded-fidelity approach (Li and Wensing, 2024) involves composing dynamics models of decreasing complexity in series within a single MPC problem. This allows planning over a long footstep horizon without the computational burden of considering a detailed model for the entire horizon. We sequence whole-body dynamics with step-to-step ALIP dynamics, providing a pathway toward real-time long-horizon footstep planning over stepping stones with whole-body

dynamics.

The remainder of the chapter is organized as follows. We first introduce the ADMM algorithm, and show how to reformulate (5.1) such that it can be solved using ADMM. We then show simulation experiments comparing the performance of ALIP MPFC using the new ADMM approach vs the previous MIQP approach. Having verified the practical efficacy of the solver, we introduce a CF-MPFC formulation which leverages this solver for sequential quadratic programming. We show simulation experiments of Cassie walking over large step-ups and step-downs using CF-MPFC. We conclude with a discussion of the remaining steps to achieve a real-time implementation of CF-MPFC.

8.1. The Alternating Direction Method of Multipliers

Inspired by recent successes in multi-contact control (Aydinoglu et al., 2023; Yang and Posa, 2024), we propose to split the MPFC problem into a convex QP and an additional set membership constraint on the decision variables (8.1). Here we review the generic recipe for solving such a problem using ADMM (Boyd et al., 2011), starting with the problem statement:

$$\underset{x}{\text{minimize}}\ f(x) \tag{8.1a}$$

$$\text{subject to } b_l \leq Ax \leq b_u \tag{8.1b}$$

$$x \in \mathcal{X}_{nc}, \tag{8.1c}$$

where $f(x)$ is a convex quadratic cost, A, b_l, and b_u form a set of linear inequality constraints, and $\mathcal{X}_{nc}$ is a potentially nonconvex set. We note that without (8.1c), (8.1) is a convex QP. We introduce a copy of the decision variables, z, to write (8.1) in a consensus formulation

$$\underset{x,z}{\text{minimize}}\ f(x) + I_{lin}(x) + I_{nc}(z) \tag{8.2a}$$

$$\text{subject to } W(x - z) = 0 \tag{8.2b}$$

where $I_{lin}(x)$ is the $0-\infty$ indicator function for the linear constraints

$$I_{lin}(x) = \begin{cases} 0, & b_l \leq Ax \leq b, \\ \infty, & \text{otherwise} \end{cases}, \tag{8.3}$$

and similarly, $I_{nc}(x)$ is the $0-\infty$ indicator function for the potentially non-convex set membership constraints

$$I_{nc}(z) = \begin{cases} 0, & z \in \mathcal{X}_{nc}, \\ \infty, & z \notin \mathcal{X}_{nc} \end{cases}. \tag{8.4}$$

The weighting matrix W is a positive definite, diagonal matrix, used to weight the norm used for the projection step in the ADMM iterations. We derive the ADMM iterations for this problem in Appendix C, reproducing them below:

$$x_{k+1} = \arg\min_x \; f(x) + I_{lin}(x) + \frac{\rho}{2}\|W(x - z_k + w_k)\|_2^2 \tag{8.5a}$$

$$z_{k+1} = \Pi^W_{\mathcal{X}_{nc}}(x_{k+1} + w_k) \tag{8.5b}$$

$$w_{k+1} = w_k + x_{k+1} - z_{k+1}, \tag{8.5c}$$

where

$$\Pi^W_{\mathcal{X}_{nc}}(v) = \arg\min_z \; (v - z)^T W^T W (v - z) \tag{8.6}$$

$$\text{subject to } z \in \mathcal{X}_{nc}. \tag{8.7}$$

When $\mathcal{X}_{nc}$ is convex, the iterates (8.5) will converge to an optimal solution, if one exists. However even when $\mathcal{X}_{nc}$ is non-convex, (8.5) has been shown empirically to generate useful solutions (Takapoui et al., 2016). In the context of real-time control, often only a small number of ADMM iterations are applied, in order to return a solution which is "good enough" on a predictable timeline. However, if the problem has not converged within the iteration limit, the set membership

constraint (8.1c) may be unacceptably violated. To account for this, we use a polishing step similar to the active-set polishing step in the OSQP solver (Stellato et al., 2020). We assume that for any $x \in \mathcal{X}_{nc}$, there is a unique maximum-volume convex polyhedron $\mathcal{C}_x$ such that $x \in \mathcal{C}_x \subseteq \mathcal{X}_{nc}$. After ADMM has reached its iteration limit, m, we solve a final convex QP representing (8.1), with (8.1c) replaced by the linear equality constraint representing $x \in \mathcal{C}_{x_{pm}}$, where $x_{pm} = \Pi^W_{\mathcal{X}_{nc}}(x_m)$. Putting this together with the standard ADMM algorithm, we arrive at Algorithm 3.

Algorithm 3 ADMM Implementation

Input: Problem Data $f, A, b_l, b_u, \mathcal{X}_{nc}$, Hyper-parameters m, ρ, W
Initialization:

$x_0 = z_0 = \arg\min\ f(x)$ s.t. $b_l \leq Ax \leq b_u$
$w_0 = 0$

for $k = 0 \dots m$ **do**
  Update x_{k+1} by solving (8.5a)
  Update z_{k+1} by the projection (8.5b)
  Update duals (8.5c)
Calculate convex restriction $\mathcal{C}_{x_{nc}}$ for $x_{nc} = \Pi^W_{nc}(x_m)$
Polishing step: $x^* = \arg\min\ f(x)$ s.t. $b_l \leq Ax \leq b_u, x \in \mathcal{C}_{x_{nc}}$
Output: Final solution x^*.

8.2. ALIP MPFC ADMM Formulation

Here we re-write the ALIP MPFC problem statement from Chapter 5 in the form Eq. (8.1), such that we can apply Algorithm 3. We then benchmark the reformulation against the original MIQP formulation.

8.2.1. Problem Reformulation

We remove the binary variables and instead enforce the stepping stone constraints directly using the set membership formulation:

$$\underset{\mathbf{x},\mathbf{p},\boldsymbol{\mu},u,T}{\text{minimize}} \; J_{mpc}(\mathbf{x},\mathbf{p}) + J_{reg}(T,u) \tag{8.8a}$$

$$\text{subject to } x_0 = A_d x_c + A A_d x_c (T - T^*) + B_d u \tag{8.8b}$$

$$x_{n+1} = A_{s2s} x_n + B_{s2s}(p_{n+1} - p_n) \tag{8.8c}$$

$$p_n \in \mathcal{P} \tag{8.8d}$$

$$\text{CoM, Input,Timing, and Footstep limits,}$$

where $\mathcal{P} = \bigcup_i \mathcal{P}_i$ is the union of the individual stepping stones. All of the constraints except (8.8d) can be formulated as linear constraints. The ADMM projection step therefore only involves a weighted projection of each footstep p_n to the stepping stone constraints. This projection is decoupled over the foothold horizon. Therefore, for M stepping stones, the projection step has a complexity of MN rather than M^N of the original mixed integer formulation. The projection for each footstep is given by

$$\Pi_{\mathcal{P}}^{W_n}(p_n), \quad i^* = \underset{p\in\mathbb{R}^3, i\in\{1\ldots M\}}{\arg\min} \; (p_n - p)^T W_n^T W_n (p_n - p) \tag{8.9a}$$

$$\text{subject to } p \in \mathcal{P}_i. \tag{8.9b}$$

For the polishing step, we construct a convex QP by replacing (8.8d) with the constraint $p_n \in \mathcal{P}_{i^*}$.

8.2.2. Implementation

Based on manual tuning, we pick the weight matrix W_n such that $W_n^T W_n = \text{diag}\left[0.3, 1.0, 0.3\right]$, weighting the y direction higher to account for Cassie's lateral underactuation. With the exception of the planning horizon, we use the same MPFC and OSC gains as the simulation experiments in Chapter 5, given in Appendix B. We grid search over the ADMM step size ρ and iteration count m to find the most successful parameters for walking over randomly placed 35 cm stepping stones (see

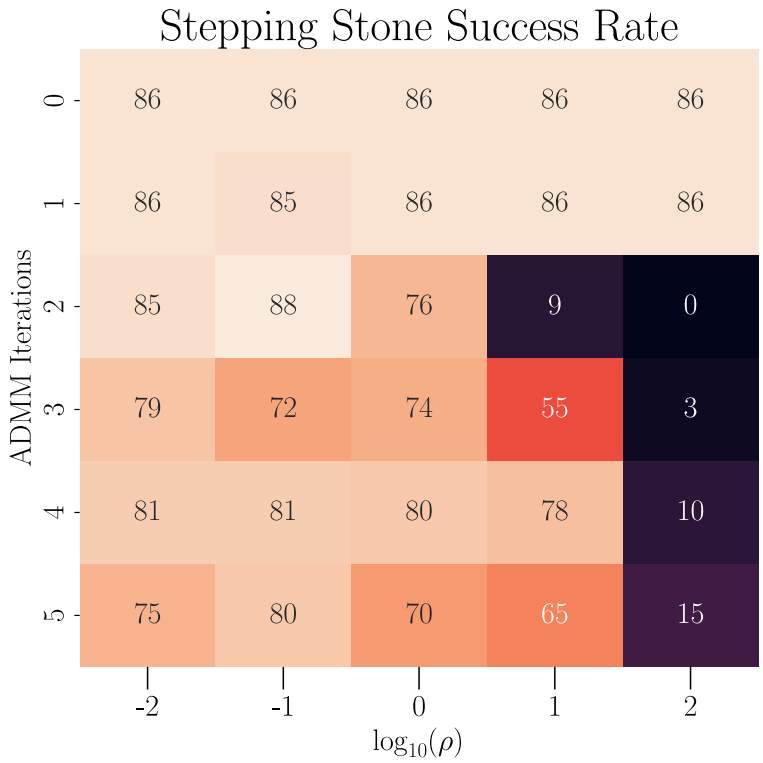


Figure 8.1: We grid search over the ADMM parameters which give the best closed-loop reliability for walking over stepping stones with ALIP MPFC. We ultimately choose $m = 0, \rho = 0.1$.

Section 5.3.1 for the detailed experimental setup) with a planning horizon of 3 footsteps (Fig. 8.1). Based on the results of the grid search, we choose $m = 0$ and $\rho = 0.1$ for our evaluation against MIQP (although ρ does not matter for this choice of m). While some some choices of ρ with multiple ADMM iterations give a slightly higher success rate, the difference is marginal, and the effect is not consistent, so we ultimately conclude it is most robust to stick with zero iterations. This corresponds to solving the MPFC problem without stepping stone constraints, projecting to the nearest footholds under the $\|\cdot\|_{W_n}$ norm, then solving the convex QP of MPFC restricted to those footholds.

The reader may wonder if ADMM with zero iterations is a reasonable choice for a local solver. Similar methods have been used successfully for quadrupedal locomotion over rough terrain such as (Grandia et al., 2022) and (Jenelten et al., 2020). These controllers define a set of desired foot placements based on the Raibert heuristic before projecting each footstep to the nearest foothold

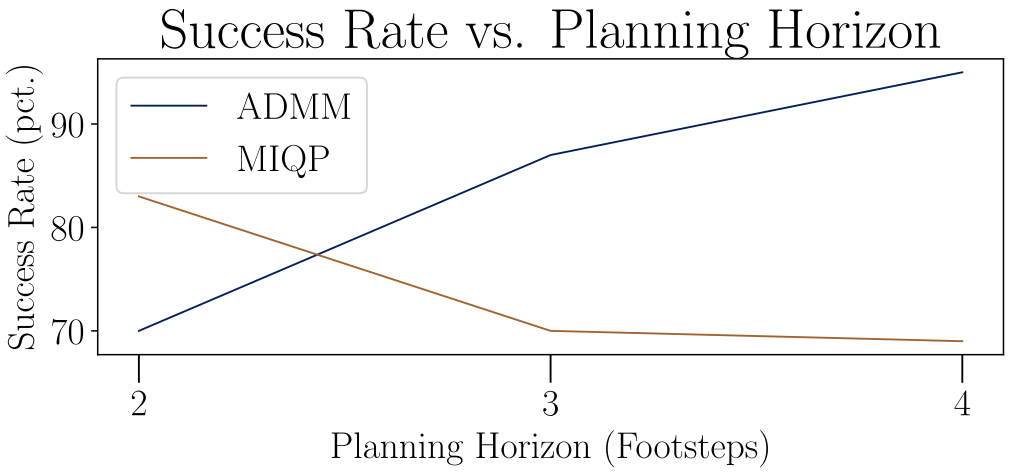


Figure 8.2: Success rate vs. planning horizon for traversing random stepping stones in simulation. We perform 100 trials, randomizing the stepping stone dimensions and positions according to the distributions in Table 5.1.

to define a convex foothold constraint for MPC. Compared to these heuristic methods, Algorithm 3 ensures that the initial footsteps are optimal with respect to the MPC problem, and provides the ability to tune the projection to account for the robot's dynamics.

8.2.3. Comparison with MIQP

We again use the stepping stone environment with $d_{min} = 35$cm to benchmark the ADMM-based solution strategy against solving an MIQP. We compare the closed loop success rate for traversing the stepping stones without falling in Fig. 8.2, noting that the MIQP solution strategy is more successful for a horizon of 2 footsteps, while ADMM is more successful than the 2-step MIQP for longer planning horizons. We also note that the success rate for the MIQP formulation decreases for longer planning horizons, falling below the success rate of ADMM for the same planning horizon. This is an unexpected result since the MIQP returns a globally optimal solution, suggesting that our cost function may be counter-productive to choosing robust foothold choices[7]. We present some hypotheses for why this might be the case, as well as a broader discussion on the trade-off between solve-time and optimality in the discussion section of this chapter. We verify that the ADMM-based approach yields faster solve times, allowing for more footsteps to be planned in real-time (Fig. 8.3).

[7]We note that solve times are not the cause of this discrepancy. Our simulation solves the MPFC problem and steps the dynamics in series, so each controller is always being solved at 100 Hz in simulated time.

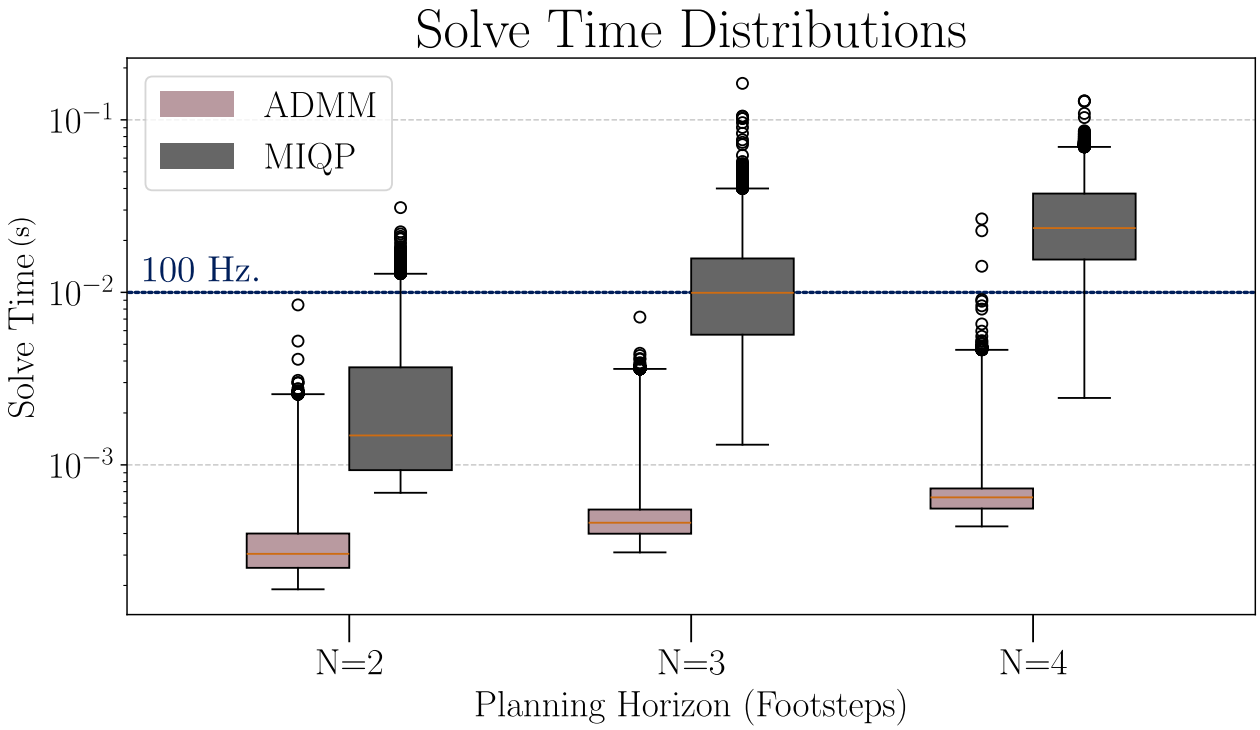


Figure 8.3: Distributions of solve times for ADMM vs. MIQP implementations of ALIP MPFC walking over random 35-40 cm stepping stones. We compile the solve times over 100 trials for each combination of solution method and planning horizon, resulting in over 100,000 data points per box plot - the exact number varies due to some simulations terminating early due to a fall. Solve times above the 99.9^{th} percentile are shown as outliers. Simulations were performed on a MacBook Pro with an Apple M1 Max CPU and 64 GB of RAM. The MIQP was solved with Gurobi using 4 threads, while ADMM was solved in a single thread, using OSQP for the ADMM iteration sub-problems.

8.3. Cascaded-Fidelity MPFC formulation

In this section, we design a cascaded fidelity MPC approach for walking over rough terrain. Note that for the examples from here forward, we will be using a fixed-spring Cassie model. Therefore we will be omitting the joints for the springs and the constraint forces which prevent their acceleration. We start by transcribing a nonlinear trajectory optimization problem formulation for whole-body MPC. We then describe how we add additional decision variables for the footstep positions and ALIP states to the whole-body trajectory optimization problem.

8.3.1. Whole-Body MPC

Here we introduce whole-body MPC as a general trajectory optimization problem statement (8.10), before describing the state-space, constraint sets, and costs in detail. We then describe our quadrature approach for transcribing (8.10) as a nonlinear program.

Trajectory Optimization Formulation

The trajectory optimization problem is given by

$$\min_{x(t),\nu(t)} \Phi(x(T)) + \int_0^T L(x(t),\nu(t),t)\,dt \tag{8.10a}$$

$$\text{subject to } \dot{x}(t) = f(x(t),\nu(t)) \tag{8.10b}$$

$$x(0) = x_0 \tag{8.10c}$$

$$\nu(t) \in \mathcal{N} \tag{8.10d}$$

$$x(t) \in \mathcal{X} \tag{8.10e}$$

$$g(x,\nu) = 0, \tag{8.10f}$$

where $x(t) = \{q(t), v(t)\}$, $\nu(t) = \{\lambda_h(t), \lambda_c(t), u(t)\}$, and the individual state and input components are as described in the Chapter 4 subsection on rigid body dynamics. The stacked contact forces are λ_c and the holonomic constraint forces from loop closures in the robot's structure are λ_h. The dynamics are given by

$$\dot{q} = N(q)v \tag{8.11}$$

$$\dot{v} = M(q)^{-1}\left(Bu + J_c(q)^T\lambda_c + J_h(q)^T\lambda_h - C(q,v)\right), \tag{8.12}$$

where $N(q)$ maps the angular velocity to the time derivative of the floating base orientation, $M(q), C(q,v), B, J_c(q), J_h(q)$ are the mass matrix, Coriolis and gravity forces, selection matrix, contact Jacobian, and holonomic constraint Jacobian.

The input constraint set is given by the friction cone constraint on the contact forces, and input bounds on the motor torques: $\mathcal{N} = \{\nu \mid \lambda_c \in \mathcal{F} \text{ and } u_{min} \leq u \leq u_{max}\}$. In general, the state

constraint set can include collision constraints between the robot and itself, and the robot and its surroundings (Chiu et al., 2022), as well as velocity limits. However for simplicity, in this work we only include joint limits: $\mathcal{X} = \{x \mid q_{min} \leq q \leq q_{max}\}$. There is also a unit norm constraint on the quaternion degrees of freedom, however we only enforce that constraint implicitly through the integration scheme to avoid over-constraining the problem.

We use the general $g(x, \nu) = 0$ constraint to encode contact constraints. This will be more naturally expressed once we have discretized the problem into a nonlinear program, but informally, these constraints are instantaneously given by the no slip condition

$$J_{c,i} v = 0 \quad \forall i \in \text{active contact points}, \tag{8.13}$$

$$\lambda_{c,i} = 0 \quad \forall i \notin \text{active contact points}. \tag{8.14}$$

The set of active contact points is time-varying. The cost function is given by (8.15), where q_{quat} is the components of the position corresponding to the floating base orientation, q are all of the positions, $\cdot_{des}(t)$ is a desired trajectory for some quantity, and $\|\cdot\|_Q^2 = (\cdot)^T Q(\cdot)$. We suppress the dependence of the state and input trajectories on t for brevity.

$$\begin{aligned} L(x, \nu, t) =& \|\text{AngleAxis}(q_{\text{quat}}, q_{quat,des}(t))\|_{q_{\text{quat}}}^2 + \|q - q_{des}(t)\|_{Q_q}^2 + \|v - v_{des}(t)\|_{Q_v}^2 \\ &+ \|\lambda - \lambda_{des}(t)\|_{Q_\lambda}^2 + \|u - u_{des}(t)\|_{Q_u}^2 \\ &+ \alpha(t) \|\psi_{sw}(q, t)\|_{Q_\psi}^2, \end{aligned} \tag{8.15}$$

where $\psi_{sw}(q, t)$ is given by

$$\max(0, \phi_{sw,des}(t) - \phi_{sw}(q, t)) \tag{8.16}$$

and $\phi_{sw}(q)$ is the signed distance between the swing foot and the terrain. This cost for the swing foot clearance avoids enforcing a maximum swing clearance, which would prevent the swing foot from crossing gaps in the terrain. The scaling factor $\alpha(t)$ decreases the swing foot clearance cost over the MPC horizon, providing freedom to explore motions which initially penetrate the terrain,

Table 8.1: Whole-Body MPC Timing Parameters

Symbol	Meaning	Value
h	time step	0.05 s
T_{ss}	single-stance duration	0.3 s
T_{ds}	double-stance duration	0.1 s
K	whole-body mpc planning horizon	16

but are refined in subsequent solves to be collision free. The terminal cost is given by

$$\begin{aligned}\Phi(x(T)) = &\|\text{AngleAxis}(q_{\text{quat}}(T), q_{quat,des}(T))\|^2_{Q_{quat,T}} \\ &+ \|q(T) - q_{des}(T)\|^2_{Q_{q,T}} + \|v(T) - v_{des}(T)\|^2_{Q_{v,T}}.\end{aligned} \tag{8.17}$$

Note that all of the costs function terms are linear or nonlinear least squares costs. We will use this in our solution strategy to efficiently form a Gauss-Newton approximation of the cost Hessian in our SQP iterations.

Nonlinear Program Formulation

Now that we have specifically enumerated the costs and constraints in our trajectory optimization, we discuss transcribing it as the nonlinear program (8.18). Our approach uses standard methods and is similar to that of Khazoom et al. (2024). The positions, velocities, and inputs are represented by piecewise quadratic, linear, and constant splines, respectively, with K segments per spline. The k^{th} spline segment has endpoints at times t_{k-1} and t_k, and the timestep is written as $h_k = t_{k+1} - t_k$. We choose the nominal h, and the single and double stance durations and such that T_{ss} and T_{ds} are both evenly divisible by h (Table 8.1). We also choose K, such that Kh is evenly divisible by $T_{ss} + T_{ds}$, which makes the model-stitching constraint more convenient to express (this is explained in the next section on cascaded-fidelity mpfc). We use trapezoidal collocation to enforce the relationship between positions and velocities, and forward Euler integration to get from accelerations to velocities. We enforce the acceleration constraints using inverse dynamics for improved computational efficiency and numerical robustness over a forward dynamics approach (Katayama and Ohtsuka, 2021). Inverse dynamics calculates the generalized forces which need to be applied to a rigid body system to produce a given generalized acceleration (e.g. $\text{ID}(q, v, \dot{v}) =$

$M(q)\dot{v} + C(q, v)$), and can be computed efficiently without explicitly constructing the mass-matrix via the Recursive Newton-Euler Algorithm (RNEA) (Luh et al., 1980). The resulting nonlinear program is given by

$$\underset{\{x_k, \nu_k\}}{\text{minimize}} \quad \Phi(x_K) + \sum_{k=0}^{K} \frac{h_k}{2}(L_k + L_{k+1}) \tag{8.18a}$$

$$\text{subject to} \quad q_{k+1} - q_k = \frac{h_k}{2}\left(N(q_k)v_k + N(q_{k+1})v_{k+1}\right) \tag{8.18b}$$

$$\text{ID}\left(q_k, v_k, \frac{v_{k+1} - v_k}{h_k}\right) = Bu_k + J_c(q_k)^T \lambda_{c,k} + J_h(q_k)^T \lambda_{h,k} \tag{8.18c}$$

$$J_h(q_k)v_k = 0 \tag{8.18d}$$

$$\gamma_{k,i} J_{c,k,i} v_k = 0 \tag{8.18e}$$

$$\text{Input, Friction, and Position Limits,} \tag{8.18f}$$

where $L_k = L(x_k, \nu_k, t_k)$. In the contact constraint (8.18e), the i subscript references to the i^{th} contact point, and $\gamma_{k,i}$ is 1 when the i^{th} contact point is in contact for knot point k, and 0 otherwise. Similarly to Khazoom et al. (2024), because the walking task is not torque limited, we only include u in the input vector for the first 3 knot points. For subsequent knot points, we enforce only the rows of (8.18c) corresponding to the un-actuated coordinates. This improves the efficiency of solving (8.18) by reducing the number of variables and constraints.

8.3.2. Cascaded Fidelity MPC Over Stepping Stones

In this section, we discuss additional decision variables, costs, and constraints we add to (8.18) to enforce stepping stone constraints and append ALIP dynamics to the end of the planning horizon. The variables and constraints for the entire problem are illustrated in Fig. 8.4.

We include decision variables for the footstep positions, p_n over a footstep planning horizon of N footsteps (indexed as $n = 0 \dots N-1$). Note that this is distinct from the whole-body planning horizon K). We include the stepping stone constraints on the footstep variables:

$$p_n \in \mathcal{P}. \tag{8.19}$$

For the footsteps which are within the whole-body planning horizon, we add a constraint that the position of the robot's swing foot at touchdown must be equal to the next stance foot position:

$$p_{sw}(q_{TD,n}, t_{TD,n}) = p_n. \tag{8.20}$$

where $p_{sw}(q,t)$ computes the position of the robot's swing foot at time t, and TD,n indexes the knot point corresponding to the n^{th} touchdown event.

For convenience, we denote the number of footsteps contained by the whole-body horizon as $F = K\frac{T_{ds}+T_{ss}}{h}$. We choose K, T_{ds}, T_{ss}, and h such that F is always an integer. We include $S = N - F + 1$ knot points for the step-to-step ALIP state, x_s^a, plus an additional copy of the ALIP state representing the mid-stance ALIP state at the final whole-body state knot point. We relate the full-order state to the ALIP state via the model-stitching constraint

$$x_c^a = g(x_K, t_K) = \begin{bmatrix} CoM_x(q_K) - p_{\text{stance},x}(q_K, t_k) \\ CoM_y(q_K) - p_{\text{stance},y}(q_K, t_k) \\ L_x(q_K, v_K, t_K) \\ L_y(q_K, v_K, t_K) \end{bmatrix} \tag{8.21}$$

$$x_0 = \exp(A(t_{TD,2} - t_k))x_c \tag{8.22}$$

where L_x and L_y are computed about p_{stance}. Because the whole-body horizon is evenly divided by the total gait cycle time, p_{stance} is always represented by the same footstep variable, simplifying the book-keeping needed to update the MPC problem data at each time-step.

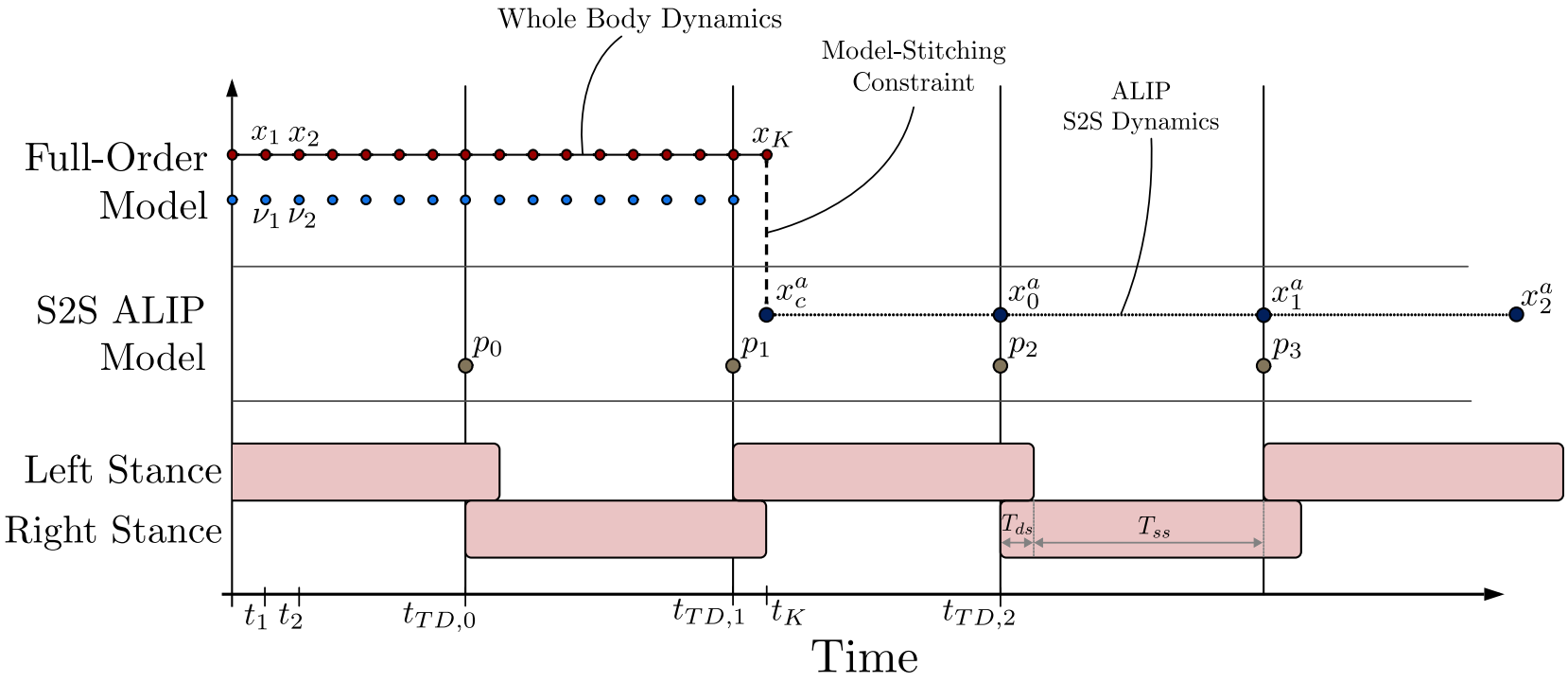


Figure 8.4: Model schedule, footstep schedule, and gait schedule for Cascaded-Fidelity MPFC.

The full Cascaded-Fidelity MPFC problem statement is given by

$$\underset{\substack{\{x_k\}_{k=0\dots K},\\ \{\nu_k\}_{k=0\dots K-1},\\ \{p_n\}_{n=0\dots N-1},\\ \{x_s^a\}_{s=0\dots S},\\ x_c^a}}{\text{minimize}} \quad \Phi(x_K) + J_{a,S}(x_S^a) + \sum_{k=0}^{K} \frac{h_k}{2}(L_k + L_{k+1}) + \sum_{s=0}^{S-1} J_{a,s}(x_s^a) + \sum_{n=0}^{N-2} J_p(p_n, p_{n+1}) \tag{8.23a}$$

$$\text{subject to} \quad q_{k+1} - q_k = \frac{h_k}{2}\left(N(q_k)v_k + N(q_{k+1})v_{k+1}\right) \tag{8.23b}$$

$$\text{ID}\left(q_k, v_k, \frac{v_{k+1} - v_k}{h_k}\right) = Bu_k + J_c(q_k)^T \lambda_{c,k} + J_h(q_k)^T \lambda_{h,k} \tag{8.23c}$$

$$J_h(q_k)v_k = 0 \tag{8.23d}$$

$$\gamma_{k,i} J_{c,k,i} v_k = 0 \tag{8.23e}$$

$$p_{sw}(q_{TD,n}, t_{TD,n}) = p_n \quad \forall n < K\frac{T_{ds} + T_{ss}}{h} \tag{8.23f}$$

$$x_c^a = g(x_K, t_K) \tag{8.23g}$$

$$x_0^a = \exp(A(t_{TD,2} - t_k)) \tag{8.23h}$$

$$x_{s+1}^a = A_{s2s} x_s^a + B_{s2s}(p_{s+F} - p_{s+F-1}) \tag{8.23i}$$

$$p_n \in \mathcal{P}, \tag{8.23j}$$

where J_a and J_p are costs on the ALIP states and footsteps, which are the same as for ALIP MPFC, given in Section 5.1.1.

Whole-Body Reference Design

For simplicity as a proof of concept, this study considers the task of forward walking. The desired orientation trajectory $q_{quat,des}(t)$ represents a constant orientation aligned with the world frame. The desired generalized velocities $v_{des}(t)$ are zero, except for the horizontal linear velocity of the floating base $v_{\text{floating base},xy}$, which is set to a constant value of the desired walking velocity. We find a reference for the joint angles, inputs, and constraint forces based on a user-specified stance width,

l, and floating base height H via inverse kinematics with a fixed-point constraint (8.24).

$$
\begin{aligned}
q^*, \lambda^*, u^* = \underset{q,\lambda,u}{\arg\min} \quad & \lambda^T \lambda + u^T u && \text{(8.24a)} \\
\text{subject to} \quad & C(q, 0) = Bu + J_\lambda^T(q)\lambda && \text{(8.24b)} \\
& q_{\text{floating base},xyz} = [0, 0, H]^T && \text{(8.24c)} \\
& q_{\text{hip yaw},L} = 0 && \text{(8.24d)} \\
& q_{\text{hip yaw},R} = 0 && \text{(8.24e)} \\
& q_{\text{quat}} = [0, 0, 0, 1]^T && \text{(8.24f)} \\
& p_{\text{right foot}}(q) = [0, -l/2, 0]^T && \text{(8.24g)} \\
& p_{\text{left foot}}(q) = [0, l/2, 0]^T && \text{(8.24h)} \\
& \phi_{\text{loop-closure}}(q) = 0, && \text{(8.24i)}
\end{aligned}
$$

where the constraint forces λ arise from Cassie's loop closures and contact forces for both feet. We then compute the desired joint trajectory as $q_{\text{joint},des}(t) = q^*_{\text{joint}}$, and the desired floating base position trajectory as

$$
q_{\text{floating base},xyz}(t) = p_{\text{stance}}(q_0, 0) + \begin{bmatrix} tv_{des,x} \\ tv_{des,y} \pm \frac{l}{2} \\ H \end{bmatrix}. \tag{8.25}
$$

The reference inputs and constraint forces are $u_{des}(t) = u^*$ and $\lambda_{des}(t) = \lambda^*$ during double stance. For single stance, we set the components of λ_{des} and u_{des} corresponding to swing leg to 0, and the components corresponding to the stance leg to the corresponding components of $2\lambda^*$ or $2u^*$.

Solution method

We represent (8.23) as a general nonlinear program with non-convex set-membership constraints,

$$\underset{z}{\text{minimize}} \quad \mathcal{J}(z,t) \tag{8.26a}$$

$$\text{subject to} \quad l(t) \leq c(z,t) \leq u(t) \tag{8.26b}$$

$$z \in \mathcal{Z}_{nc} \tag{8.26c}$$

Where the dependence of the costs and constraints on t represents the parameterization of the problem based on the current time with-respect to the nominal gait switching pattern. For the i^{th} control iteration, we use the current solution z_i to solve one SQP iteration of the form (8.27), using the ADMM solver in Algorithm 3 to find a search direction δ_i:

$$\delta_i = \underset{\delta}{\arg\min} \quad \frac{1}{2}\delta^T \nabla_z^2 \mathcal{J}(z_i,t)\delta + \nabla_z \mathcal{J}(z_i,t)^T \delta \tag{8.27a}$$

$$\text{subject to} \quad l(t) \leq \nabla_z c(z_i,t)^T \delta \leq u(t) \tag{8.27b}$$

$$z_i + \delta \in \mathcal{Z}_{nc}. \tag{8.27c}$$

The cost hessian $\nabla_z^2 \mathcal{J}(z,t)$ is the Gauss-Newton Hessian to ensure that the cost function for the SQP subproblems is positive-semidefinite. For each individual cost term, the Guass-Newton hessian is given by

$$\nabla_z^2 \left(\frac{1}{2} \|r(z_i)\|_W^2 \right) = R(z_i)^T W R(z_i), \tag{8.28}$$

where $R(z) = \frac{\partial r}{\partial z}$ is the Jacobian of the residual r. The Guass-Newton Hessian for the full cost is then the sum of the Hessians of the individual cost terms. After solving (8.27) for a search direction, we update z via

$$z_{i+i} = z_i + \beta_i \delta_i \tag{8.29}$$

where β_i is computed via the filter line-search proposed in (Grandia et al., 2022).

Implementation Details

We solve the SQP subproblems using the same ADMM parameters and footstep constraint projection weights as the ALIP MPFC example. Before each solve, we adapt the timestep size, h_k, during the initial stance phase, such that the knot points are evenly spaced, and contact transitions occur exactly at the knot points. The cost weights and planning horizon details for our implementation are given in Appendix D.

Because CF-MPFC plans over whole-body dynamics, the resulting solutions are already dynamically feasible, so we simply track the solution using PD control with feedforward torques at the actuated joints:

$$u_{cmd}(t) = u_{sol}(t) + K_p(q_{sol}(t) - q) + K_d(v_{sol}(t) - v) \tag{8.30}$$

where $K_p \in \mathbb{R}^{n_u \times n_q}$ and $K_d \in \mathbb{R}^{n_u \times n_v}$ are proportional and derivative gain matrices which select the actuated coordinates from the position and velocity error vectors. A comparison of the overall control architectures for ALIP MPFC and CF-MPFC is given in Fig. 8.5.

8.4. Evaluation

This section gives a proof-of-concept evaluation of CF-MPFC. First we show that CF-MPFC is able to traverse larger height changes than ALIP MPFC, then we profile the CF-MPFC solver iterations to determine future steps toward a real-time implementation.

8.4.1. Height Change Test

Cassie traverses 20, 25, and 30 cm height changes in simulation in one continuous trial (Fig. 8.7). Because the swing foot planner does not account for the robot's dynamics, ALIP MPFC is not able to traverse the same terrain, as the swing foot does not make it all the way onto the 20 cm block, and Cassie slips off (Fig. 8.6). For completeness, we also show CF-MPFC traversing the same stepping stones, narrow beam, and stairs as ALIP MPFC was demonstrated on in Section 5.3.1 (Fig. 8.8).

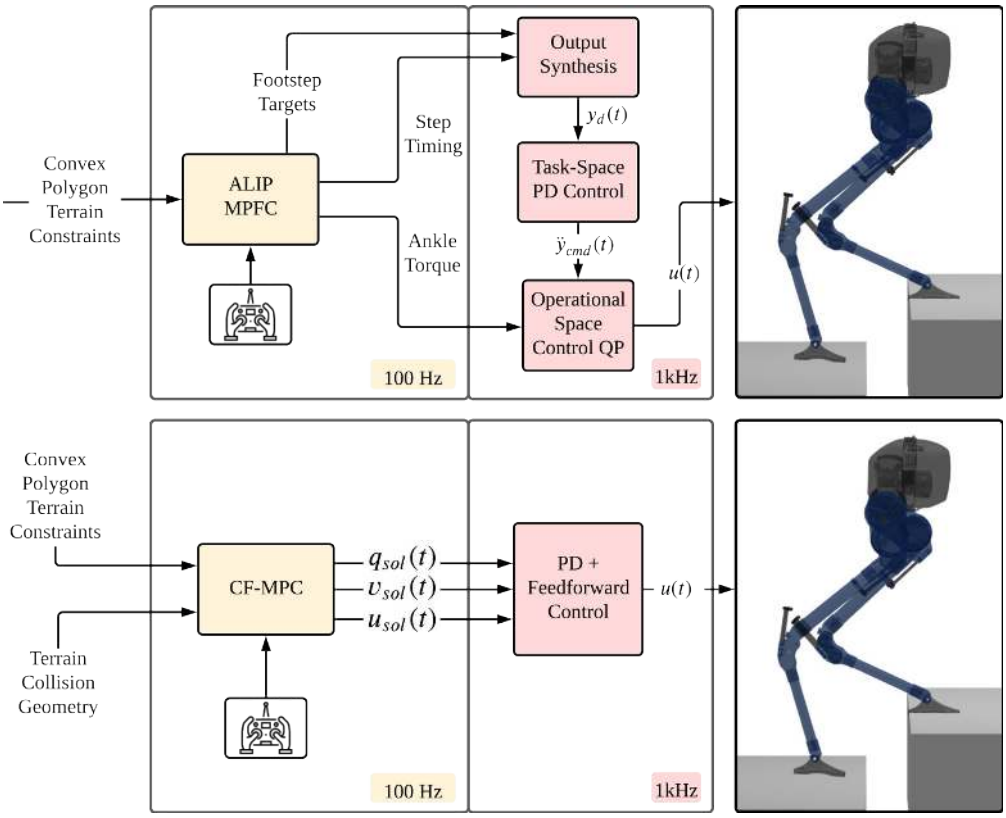


Figure 8.5: Comparison of the control architectures for ALIP MPFC (top) and CF-MPFC (bottom).

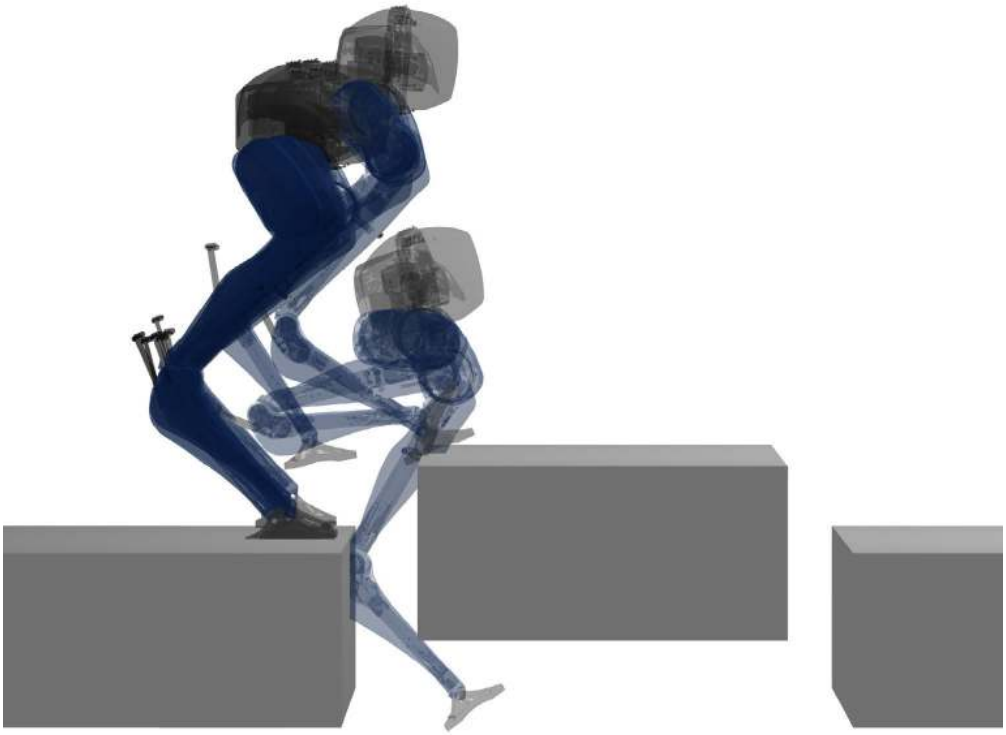


Figure 8.6: ALIP MPFC fails to complete a 20 cm step up due to the swing foot not making it on to the step. Relying on a heuristic swing foot planner places a high burden on the OSC tracking performance compared to CF-MPFC, which accounts for the robot's dynamics when planning a swing foot motion.

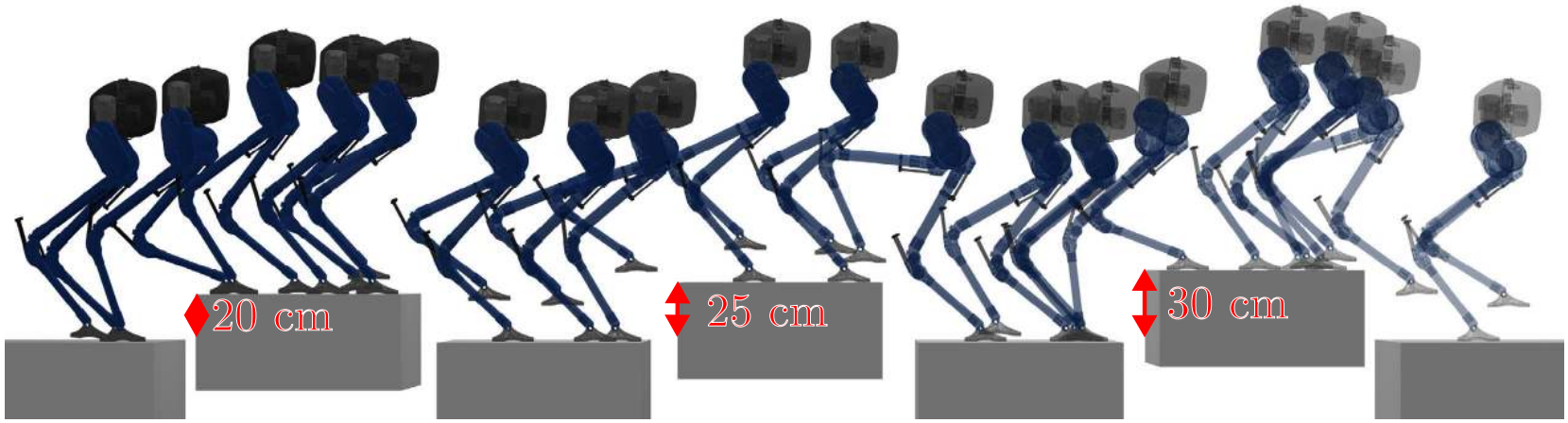


Figure 8.7: Cassie successfully traversing up to 30 cm height changes using CF-MPFC in simulation.

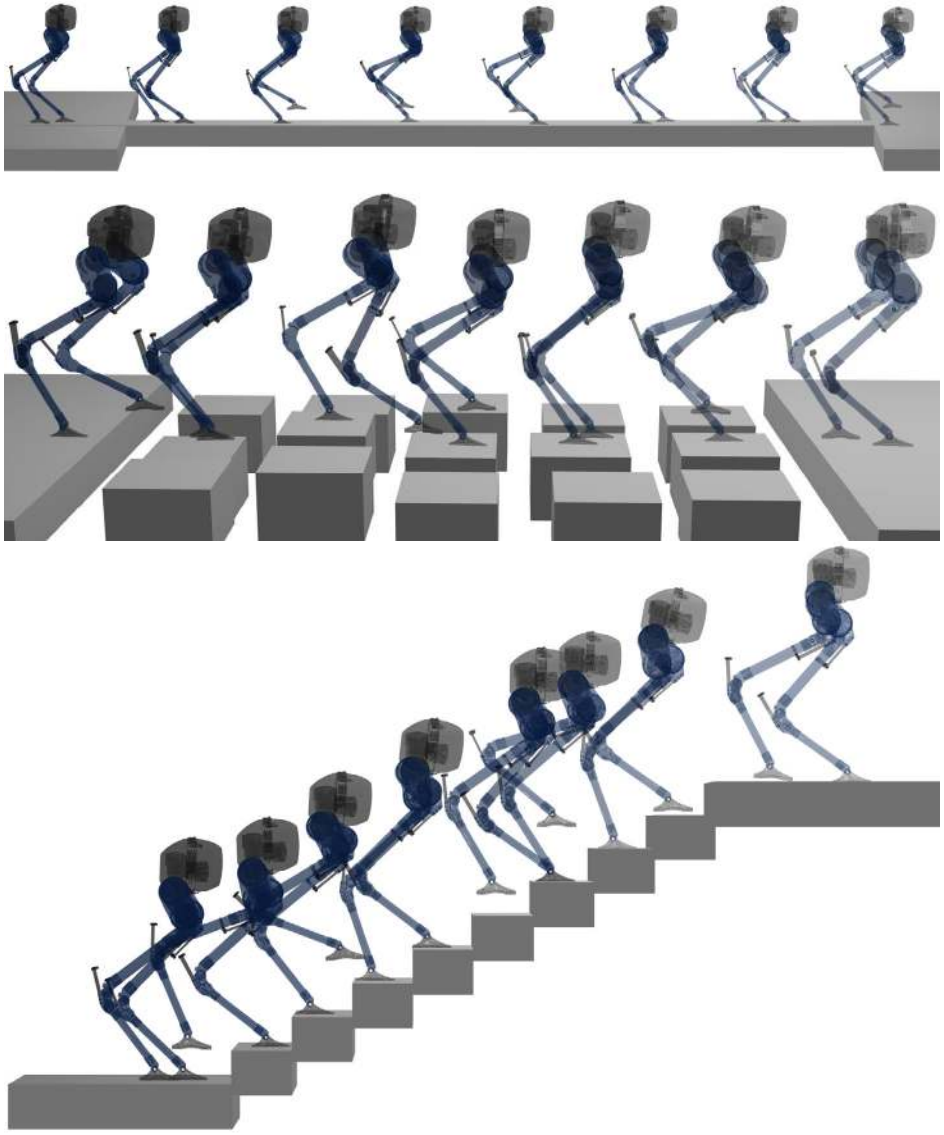


Figure 8.8: Cassie traverses a narrow beam, stepping stones, and stairs using CF-MPFC.

8.4.2. Profiling

Our implementation of CF-MPFC is not yet solvable in real time. We break-down the SQP iteration computation time by sub-task in Table 8.2. The data is from walking over the tall boxes in Fig. 8.7. The controller is running on a MacBook Pro with a 10-core Apple M1 Max CPU and 64 GB of RAM. To reach a 70 Hz. solve time, we need to decrease the total solve-time by approximately one order of magnitude as well as reduce jitter.

Table 8.2: Computation Time Breakdown for CF-MPFC Iterations

Subtask	Setup QP	Solve QP	Line Search	Total
Mean Computation time (s)	0.023	0.115	0.002	0.139
Std.	0.002	0.044	0.000	0.044
Max.	0.038	0.179	0.003	0.217
Min.	0.020	0.026	0.001	0.049

Most of the computation time comes from two factors. First, we use finite-differencing to take the gradients of the nonlinear constraints. This is not only slow, but ignores advantageous sparsity structure, increasing the solve time of the resulting QP. We provide analytical gradients for most of the cost components, which is faster than finite differencing, but still potentially ignores sparsity. Similar methods generally use Pinocchio (Carpentier et al., 2019) with automatic differentiation or code generation tools (e.g. Casadi (Andersson et al., 2019)) to quickly generate sparse gradients of the nonlinear costs and constraints.

More challenging is the fact that we use relatively slow OSQP settings to reliably solve the polishing step QP in Algorithm 3. To ensure that the foothold constraints are not violated, we require OSQP to perform up to 1000 iterations, and we leave ρ-scaling enabled, which requires additional matrix factorizations. Future work will consider alternate solvers, and attempt to scale the constraints such that the QP solve times are competitive with state-of-the-art. Khazoom et al. (2024), for example, cite using 20 OSQP iterations with a fixed ρ to achieve a 90 Hz update rate with 32 knot points.

8.5. Discussion

This chapter explores handling challenging optimal control problems with local methods. We leverage ADMM for handling stepping stone constraints, and sequential quadratic programming for handling nonlinear dynamics and cost functions. Because we lack optimality guarantees for such methods, we rely on empirical validation, which has the limitation of unclear generalization beyond the test conditions. Although, as seen from Fig. 8.2, the trade-off between computational speed (via local solutions) and optimality is not necessarily straight-forward. The fact that the globally optimal solution leads to worse closed-loop behavior suggests that our cost function, which prioritizes velocity tracking, is not the correct choice for robust foothold selection. Future work could

look at the bias imposed on foothold selection by the ADMM approach and attempt to capture this in a convex quadratic cost term. Additionally, for Algorithm 3, future work involves characterizing under what conditions the polishing step is infeasible, and if there are easy fall-back methods for repairing an infeasible stepping stone sequence.

Local methods are also inherently less exploratory. This is especially relevant for CF-MPFC, where we are using a local solution to address the challenges posed by both combinatorial foothold choices and nonlinear dynamics. For example, at lower commanded velocities, CF-MPFC struggles to find solutions which make forward progress when approaching tall step-ups. The robot either gets too close to the obstacle before finding a solution which steps onto it, ultimately tripping, or the swing foot penalty pushes it away from the obstacle and it steps in place indefinitely.

From an implementation perspective, both ALIP MPFC and CF-MPFC required a large number of maximum iterations in the polishing step QP to reliably satisfy the foothold constraints, and using a different solver than OSQP may provide better performance.

8.6. Conclusion

In this chapter, we provide a local solver based on ADMM which eliminates the combinatorial complexity of MPFC, ultimately enabling real-time footstep planning with a horizon of four footsteps instead of two, increasing reliability over a challenging stepping stone task.

We apply this solver to the sequential quadratic programming subproblems of a cascaded-fidelity MPFC formulation. By combining whole-body MPC with long-horizon footstep planning over constrained footholds, we provide a proof of concept for an exceedingly general-purpose walking controller. In addition to the solve-time improvements discussed in Section 8.4.2, future work can consider including gait-timing adaptation and self collision constraints to further improve the robustness of the controller and enable behaviors such as cross-over stepping.

CHAPTER 9

Conclusions and Future Work

In this thesis, we address the challenge of underactuated bipedal walking over rough terrain, providing real-time algorithms for perception and control which respectively exceed the robustness and generality of previous model-based approaches. This chapter briefly recapitulates our specific contributions before offering some broader opinions and discussion of potential future work.

In Chapter 5, we introduce Model Predictive Footstep Control, an MPC-style reactive footstep planner for underactuated walking over stepping stones, which jointly optimizes over reduced order model dynamics, footstep plans, and the discrete choice of stepping surface. We formulate MPFC as a single MIQP which can be solved at over 100 Hz. to stabilize walking over discontinuous terrain without a pre-specified foothold sequence.

In Chapter 6, we argue that plane segmentation, the most popular choice for identifying safe terrain from vision, is fundamentally brittle. We introduce an algorithm which is structurally similar to existing approaches but which omits a plane segmentation step, and show that our algorithm is faster and more consistent in classifying the steppable terrain. We develop a convex decomposition pipeline for converting the resulting steppability mask into a set of convex polygons representing the safe terrain.

Chapter 7 ties together the contributions from Chapter 5 and Chapter 6 into the first hardware demonstration of Cassie traversing discontinuous terrain with model-based control.

Finally, Chapter 8 increases the robustness and generality of MPFC specifically. First, we increase the robustness of MPFC by planning over longer footstep horizons in real time using ADMM. We show that longer planning horizons with a locally optimal solver yield a higher closed-loop success rate than a globally optimal short horizon plan generated through the MIQP formulation. Second, we increase the height changes traversable by Cassie using a cascaded-fidelity MPC formulation which sequences whole-body dynamics with step-to-step ALIP dynamics. We formulate a single

optimization problem over the stepping stone choice, whole-body dynamics, and footstep positions, providing a discussion of necessary steps for a real-time implementation.

9.1. Lessons Learned

At time of writing, the field is continuing to move from MPC toward sim-to-real RL as a popular approach for legged locomotion. While MPC will likely remain a key approach in the near and medium term, we should ask what insights can we draw from this thesis that are durable to such trends, in that they are applicable to a wide range of methods and problem settings.

In the broader context of designing robotics systems for deployment in unstructured environments, there are three major lessons. First, modeling choices, and how they are interpreted for the purpose of algorithm design, have an out-sized impact on algorithm performance, both in terms of speed and reliability. Our work on S3 most clearly emphasizes this point - we improve the terrain segmentation speed and robustness by recognizing that piecewise planar terrain is a convenient assumption for numerical optimization, not a prescription for how to design a terrain segmentation algorithm. For the foreseeable future, roboticists will have to think carefully about how information is used across an entire system to design appropriate subsystems for sensing or acting on that information.

Second, while general purpose solvers are convenient for prototyping algorithms, many optimization problems encountered in robotics have some structure which makes them easier to solve than a general instance of their problem class. In this context, it is often relatively easy to write a solver which outperforms general purpose methods on these specific problems. This fact has been leveraged extensively across robotics, especially in optimization-based control (e.g. Kuindersma et al. (2014); Nguyen et al. (2024)), and simulating contact between rigid bodies (Castro et al., 2023; Le Cleac'h et al., 2023). In this thesis, we leverage custom solvers for fast whole-body control using FCCQP (Acosta, 2024), making cut selections in the whittling algorithm, and using ADMM for MPFC in Chapter 8. A concrete recommended first step, especially for small problems with demanding solve time requirements, is to write out the optimality requirements, and see if a simple closed form or iterative approach presents itself.

Finally, for complicated systems such as a perceptive locomotion stack, often the only way to identify the most impactful research questions is to run hardware experiments. This can be accomplished by starting with a best guess for how to do it based on the state of the art, and recording enough data from each subsystem to diagnose failure points. Before we started our work on perceptive locomotion, we assumed that we would be able to use an off-the-shelf solution for plane segmentation, and that the control design would be the hard part. This did not turn out to be the case. Our first iteration of MPFC, while not as effective as the final version presented here, was already somewhat functional and represented increased capabilities over the state of the art. It took another year and a half and some new ideas to get the perception system in a state where Cassie could walk continuously with vision in the loop. Hopefully, this thesis can serve as a reference for getting a baseline implemented quickly, and outlining some of the hazards which lay in the path. Good luck!

9.2. Future Work

MPFC was one of several early works for solving the stepping stone problem with dynamics constraints in real time, and the field continues to push capabilities in this direction (Zhou et al., 2025). That said, we impose significant structure in order to reach the real-time requirements needed for hardware deployment. We assume a linear reduced order model, convex quadratic costs, and a convex polygon terrain representation. In addition to pushing our whole-body MPFC implementation into real time, we should also consider whether a nonlinear or non-convex cost in the ALIP MPFC problem would lead to more robust foothold selections. Recent progress in sampling-based MPC (Xue et al., 2024; Howell et al., 2022) suggests that a sampling-based approach may be effective for optimizing over non-convex costs, and planning footsteps directly on the steppability segmentation.

While the contributions of this thesis expand the capabilities of underactuated bipeds to include discontinuous terrain, low level challenges impact the physical system's robustness and ability to traverse more challenging terrains such as conventional stairs. This is especially relevant to the model's assumption of which points on the robot are in no-slip contact with the ground. Our state estimator assumes no-slip contact, leading to position and velocity estimation errors when the robot does slip, which propagate to the elevation mapping and control systems. Further research

is needed in general into state estimation for bipedal robots which is robust to the contact state. Incorporating visual information can be a source of this robustness, albeit with a non-trivial increase in the implementation effort required (Wisth et al., 2023). Along this line, MPC formulations should be developed which are robust to the nominal contact mode, either through detection and adaptation of the actual contact configuration, or by specifying desired behavior which is open-loop robust to variations in the contact timing and configurations. This could also include the ability to recognize and leverage partial footholds and line contacts for traversing the most challenging terrains. This type of robustness and full leveraging of the dynamics is often associated with sim-to-real RL (e.g. Miki et al. (2022a); Siekmann et al. (2021)), although finding approaches which maintain these qualities while achieving the precision of model-based footstep planning remains an open question.

Ultimately, by introducing a safe terrain segmentation algorithm which is robust and easy to implement, and extending state-of-the-art model-based controllers to handle discrete footholds, this thesis provides a concrete starting point for addressing these problems inside of a full-stack hardware implementation.

APPENDIX A

ALIP MPFC IMPLEMENTATION DETAILS

A.1. ALIP S2S Lateral Reset Map Adjustment

Because B_{ds} is decoupled in x and y in (4.19), our MPFC implementation assumes $f(t) = 1$ for the lateral components of the ALIP state, which corresponds to instantaneous weight transfer at the beginning of double-stance. In this case, (4.19) evaluates to

$$B_{ds} = A^{-1}(A_r - I)B_{CoP}. \tag{A.1}$$

This helps the robot more closely track the desired step width by compensating for systematic error in swing foot tracking. The robot consistently steps with a wider step width than the commanded footstep position. Due to compliance and backlash in Cassie's hip roll joints, we are unable to raise the PD gains on the lateral swing foot position beyond the values in Table B.2. Adjusting the reset map to assume instantaneous weight transfer acts as a feedforward correction by increasing the model's estimate of how much momentum will be absorbed by a larger lateral footstep size. To calculate the final value of B_{ds}, we take the inner 2×2 submatrix, which corresponds to the coronal plane, from (A.1), and the 4 corner values from (4.20), which correspond to the sagittal plane.

A.1.1. Constructing the Desired-Velocity Subspace for MPFC

Here we derive the affine subspace of P2 orbits which achieve a given desired velocity v_{des}. We will define the projection matrices Π_0 and Π_1, and the offsets d_0 and d_1, and then for the general case, we have that

$$\Pi_{n+2} = \Pi_n$$

$$d_{n+2} = d_n$$

We start by unrolling the s2s dynamics over two footsteps, and applying the P2 orbit constraint $x_2 = x_0$:

$$x_0 = A_{s2s}^2 x_0 + A_{s2s} B_{s2s} \delta p_0 + B_{s2s} \delta p_1. \tag{A.2}$$

We substitute the velocity constraint $(\delta p_0 + \delta p_1) = 2T_{s2s} v_{des}$ into (A.2) and solve for x_0:

$$x_0 = G(A_{s2s} B_{s2s} - B_{s2s}) \delta p_0 + 2T_{s2s} G B_{s2s} v_{des}, \tag{A.3}$$

where $G = (I - A_{s2s}^2)^{-1}$. From (A.3) we have a definition of the desired velocity subspace as an offset based on v_{des} and the span of $L_0 = G(A_{s2s} - I)B_{s2s} \in \mathbb{R}^{4\times 2}$. We convert this to the desired form (5.3) by left-multiplying with Π_0, a projection matrix to the orthogonal complement of the range of L_0. Because Π_0 maps $L_0 \delta p_0$ to zero for any δp_0 by construction, this leaves us with

$$\Pi_0 x_0 = 2\Pi_0 T_{s2s} G B_{s2s} v_{des}, \tag{A.4}$$

so $d_0(v_{des}) = 2T_{s2s} G B_{s2s} v_{des}$. To find Π_1, we use

$$\begin{aligned} x_1 &= A_{s2s} x_0 + B_{s2s} \delta p_0 \\ \therefore\ x_1 &= A_{s2s}(L_0 \delta p_0 + d_0) + B_{s2s} \delta p_0 \\ \therefore\ x_1 &= (A_{s2s} L_0 + B)\delta p_0 + A_{s2s} d_0 \\ \therefore\ L_1 &= A_{s2s} L_0 + B, d_1 = A_{s2s} d_0. \end{aligned} \tag{A.5}$$

Π_1 is similarly constructed as a projection to $\text{span}(L_1)^\perp$.

APPENDIX B

MPFC AND S3 PARAMETERS

The following tables give the parameters used for each component of our stack. Diagonal matrices are represented as d[$\cdots$], where the arguments to d represent the entries on the diagonal of the matrix. While we did not extensively tune MPFC costs, we found it worked best to set Q_N at least $100\times$ larger than Q for the position coordinates. This avoided short-sighted behavior when walking over edges, and could maybe be reduced for longer planning horizons.

Compared to hardware, our simulation experiments feature increased MPFC state costs and OSC PD gains, and decreased S3 resolution and hysteresis. These differences showcase the full potential of our method with more-precise swing-foot tracking. We switch the inpainting approach from Navier-Stokes (NS) to Least-Neighboring-Value (LNV) to align with the discrete terrain tested in sim.

Table B.1: MPFC Parameters

+ **Symbol**	**Meaning**	**Hardware**	**Simulation**
N	MPFC Horizon	2 steps	2 steps
t_{min}	Min. SS duration	0.27 s	0.27 s
t_{max}	Max. SS duration	0.33 s	0.33 s
H	ALIP height	0.85 m	0.85 m
T_{ss}	Nominal SS duration	0.3 s	0.3 s
T_{ds}	DS duration	0.1 s	0.1 s
w_T	Time weight	100	100
l	Step width	0.2 m	0.15 m
w_u	Ankle torque weight	0.01	0.01
u_{max}	Max. ankle torque	22 Nm	22 Nm
Q_N	Terminal state cost	d[100, 100, 1, 1]	d[1000, 1000, 20, 20]
Q	Runnning state cost	d[0.001, 0.1, 0.01, 0.001]	d[10, 10, 5, 5]
R	Running step size cost	d[25, 25, 0]	d[25, 25, 0]
–	CoM soft pos. limits	± [0.35, 0.35] m	± [0.4, 0.4] m
–	CoM soft vel. limits	± [2.5, 1.5] m/s	± [2.5, 1.5] m/s
–	Soft constraint cost	1000	1000

Table B.2: OSC Gains (Hardware)

OSC Objective	**W**	**Kp**	**Kd**
Toe joint angle	1	1500	10
Hip yaw angle	2	40	2
CoM [x, y, z]	[0, 0, 10]	[0, 0, 100]	[0, 0, 6]
Pelvis [roll, pitch, yaw]	[2, 4, 0.02]	[200, 200, 0]	[10, 10, 4]
Swing Foot [x, y, z]	[4, 4, 2]	[220, 180, 180]	[6, 5.5, 5.5]
Ankle Torque	10	–	–

Table B.3: OSC Gains (Simulation)

OSC Objective	**W**	**Kp**	**Kd**
Toe joint angle	1	1500	10
Hip yaw angle	2	100	4
CoM [x, y, z]	[0, 0, 10]	[0, 0, 80]	[0, 0, 10]
Pelvis [roll, pitch, yaw]	[2, 4, 0.02]	[200, 200, 0]	[10, 10, 10]
Swing Foot [x, y, z]	[4, 4, 2]	[400, 400, 400]	[20, 20, 25]
Ankle Torque	10	–	–

Table B.4: Perception Stack Parameters

Symbol	**Meaning**	**Hardware**	**Simulation**
	Elevation Mapping		
–	Map Size	3×3 m	2.5×2.5 m
–	Map Resolution	0.03 m	0.025 m
	S3		
k_{hyst}	Safety Hysteresis	0.6	0.4
k_{safe}	Safety Threshold	0.7	0.7
–	Safety Margin Kernel Size	4 px	4 px
σ_{LoG}	LoG Standard Dev. for c_{curve}	2 px	2 px
α_c	c_{curve} scaling parameter	5	5
–	c_{inc} kernel size	5 px	5 px
–	Inpainting	NS	LNV
	Convex Decomposition		
d	ACD Concavity Limit	0.25 m	0.25 m

APPENDIX C

ADMM DERIVATION WITH WEIGHTED PROJECTION

This appendix derives the somewhat non-standard ADMM iterations used in Chapter 8

$$\underset{x,z}{\text{minimize}}\ f(x) + I_{lin}(x) + I_{nc}(z) \tag{C.1a}$$

$$\text{subject to } W(x - z) = 0. \tag{C.1b}$$

Letting y be the dual variable associated with the $W(x - z) = 0$ constraint, the augmented Lagrangian for (C.1) is

$$L_\rho(x, z, y) = f(x) + I_{lin}(x) + I_{nc}(z) + y^T W(x - z) + \frac{\rho}{2}\|W(x - z)\|_2^2. \tag{C.2}$$

The ADMM iterations are then given by

$$x_{k+1} = \arg\min_x\ f(x) + I_{lin}(x) + y_k^T W(x - z_k) + \frac{\rho}{2}\|W(x - z_k)\|_2^2 \tag{C.3}$$

$$z_{k+1} = \arg\min_z\ I_{nc}(z) + y_k^T W(x_{k+1} - z) + \frac{\rho}{2}\|W(x_{k+1} - z)\|_2^2 \tag{C.4}$$

$$y_{k+1} = y_k + \rho W(x_{k+1} - z_{k+1}) \tag{C.5}$$

We introduce a scaled dual variable w such that $y = \rho W w$. We rewrite the augmented Lagrangian using the scaled variable

$$L_\rho(w, z, w) = f(x) + I_{lin}(x) + I_{nc}(z) + \rho w^T W^T W(x - z) + \frac{\rho}{2}\|W(x - z)\|_2^2 \tag{C.6}$$

$$= f(x) + I_{lin}(x) + I_{nc}(z) + \frac{\rho}{2}\|W(x - z + w)\| - \frac{\rho}{2}\|Ww\|_2^2 \tag{C.7}$$

Additionally, the dual update is given as

$$\rho W w_{k+1} = \rho W w_k + \rho W(x_{k+1} - z_{k+1}). \tag{C.8}$$

Because W is positive definite, we can simplify this to $w_{k+1} = w_k + x_{k+1} - z_{k+1}$. This gives the the following ADMM iterations, where constant terms have been omitted from each optimization problem:

$$x_{k+1} = \arg\min_x \ f(x) + I_{lin}(x) + \frac{\rho}{2}\|W(x - z_k + w_k)\|_2^2 \tag{C.9}$$

$$z_{k+1} = \arg\min_z \ I_{nc}(z) + \frac{\rho}{2}\|W(x_{k+1} - z + w_k)\|_2^2 \tag{C.10}$$

$$w_{k+1} = w_k + x_{k+1} - z_{k+1}. \tag{C.11}$$

The z update can be expressed as the following weighted projection, where $d_k = x_{k+1} - w_k$

$$z_{k+1} = \arg\min_z \ (d_k - z)^T W^T W (d_k - z) \tag{C.12a}$$

$$\text{subject to } z \in \mathcal{X}_{nc}. \tag{C.12b}$$

APPENDIX D

CASCADED FIDELITY MPFC GAINS

This section provides the parameters for the CF-MPFC implementation. Successfully traversing tall obstacles was somewhat sensitive to the floating base position costs, however the same costs transferred to the beam, stairs, and stepping stones.

Table D.1: CF-MPFC Parameters

Symbol	**Meaning**	**Value**
K	Number of Knot Points	16
h	Timestep	0.05
–	Full-torque knots	3
μ	Friction Coefficient	0.6
H	Desired Height	0.95 m
l	stance width	0.2 m
N	Total Footsteps	4
T_{ss}	single-stance duration	0.3 s
T_{ds}	double-stance duration	0.4 s
Q_ψ	swing clearance cost-weight	4000

In Table D.2, the order of the positions is

$$\begin{bmatrix} q_{\text{quat},w}, q_{\text{quat},x}, q_{\text{quat},y}, q_{\text{quat},z}, q_{\text{floating base},x}, q_{\text{floating base},y}, q_{\text{floating base},z} \\ q_{\text{hip roll},L}, q_{\text{hip yaw},L}, q_{\text{hip pitch},L}, q_{\text{knee},L}, q_{\text{ankle},L}, q_{\text{toe},L} \\ q_{\text{hip roll},R}, q_{\text{hip yaw},R}, q_{\text{hip pitch},R}, q_{\text{knee},R}, q_{\text{ankle},R}, q_{\text{toe},R} \end{bmatrix}$$

and the order of the velocities is

$$\begin{bmatrix} v_{\omega x}, v_{\omega y}, v_{\omega z}, v_{\text{floating base},x}, v_{\text{floating base},y}, v_{\text{floating base},z} \\ v_{\text{hip roll},L}, v_{\text{hip yaw},L}, v_{\text{hip pitch},L}, v_{\text{knee},L}, v_{\text{ankle},L}, v_{\text{toe},L} \\ v_{\text{hip roll},R}, v_{\text{hip yaw},R}, v_{\text{hip pitch},R}, v_{\text{knee},R}, v_{\text{ankle},R}, v_{\text{toe},R} \end{bmatrix}$$

Table D.2: CF-MPFC Whole-Body Cost Weights

Position Weights	
Q_q	diag[1e-4, 1e-4, 1e-4, 1e-4, 2, 2, 8, 0.01, 5, 3.5, 4.0, 0.8, 1, 0.01, 5, 3.5, 4.0, 0.8, 1]
Q_{quat}	[1, 1, 1]
Velocity Weights	
Q_v	diag[1, 0.2, 5, 1.0, 0.1, 0.1, 0.001, 0.01, 0.01, 0.001, 1e-4, 0.01, 0.001, 0.01, 0.01, 0.001, 1e-4, 0.01]
Lambda Weights	
W_λ	diag[1e-4, 1e-4, 1e-4, 1e-4, 1e-5, 1e-4, 1e-4, 1e-5, 1e-4, 1e-4, 1e-5, 1e-4, 1e-4, 1e-5]
Input Weights	
W_u (Hip roll) (Hip yaw) (Hip pitch) (Knee) (Toe)	diag[0.00005, 0.00005, 0.001, 0.001, 0.001, 0.001, 1e-4, 1e-4, 0.001, 0.001]
Final Weights	
$Q_{q,T}$	diag[0.01, 0.01, 0.01, 0.01, 1, 1, 10, 1, 10, 1, 0.1, 0.1, 1, 1, 10, 1, 0.1, 0.1, 1]
$Q_{quat,T}$	diag[5, 5, 10]
$Q_{v,T}$	diag[1, 1, 1, 4, 4, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Table D.3: CF-MPFC ALIP cost weights

Q	$4I_{4\times4}$
Q_N	$10I_{4\times4}$
R	diag[15, 15, 0]

BIBLIOGRAPHY

Bernardo Aceituno-Cabezas, Carlos Mastalli, Hongkai Dai, Michele Focchi, Andreea Radulescu, Darwin G. Caldwell, José Cappelletto, Juan C. Grieco, Gerardo Fernández-López, and Claudio Semini. Simultaneous Contact, Gait, and Motion Planning for Robust Multilegged Locomotion via Mixed-Integer Convex Optimization. *IEEE Robotics and Automation Letters*, 3(3):2531–2538, July 2018. ISSN 2377-3766. doi: 10.1109/LRA.2017.2779821.

Brian Acosta. FCCQP: A Whole Body Control QP Solver with Full Friction Cones, 2024. URL https://github.com/Brian-Acosta/fcc_qp/.

Brian Acosta and Michael Posa. Bipedal Walking on Constrained Footholds with MPC Footstep Control. In *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pages 1–8, December 2023. doi: 10.1109/Humanoids57100.2023.10375170.

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, 2019. doi: 10.1007/s12532-018-0139-4.

Taylor Apgar, Patrick Clary, Kevin Green, Alan Fern, and Jonathan Hurst. Fast online trajectory optimization for the bipedal robot cassie. In *Robotics: Science and Systems*, volume 101, 2018.

Alp Aydinoglu, Adam Wei, and Michael Posa. Consensus Complementarity Control for Multi-Contact MPC, April 2023.

Hamid Benbrahim and Judy A. Franklin. Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems*, 22(3):283–302, 1997. ISSN 0921-8890. doi: https://doi.org/10.1016/S0921-8890(97)00043-2. URL https://www.sciencedirect.com/science/article/pii/S0921889097000432. Robot Learning: The New Wave.

M. Bertalmio, A.L. Bertozzi, and G. Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001. doi: 10.1109/CVPR.2001.990497.

Sylvain Bertrand, Inho Lee, Bhavyansh Mishra, Duncan Calvert, Jerry Pratt, and Robert Griffin. Detecting Usable Planar Regions for Legged Robot Locomotion. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4736–4742, October 2020. doi: 10.1109/IROS45743.2020.9341000. URL https://ieeexplore.ieee.org/abstract/document/9341000. ISSN: 2153-0866.

Teng Bin, Jianming Yao, Tin Lun Lam, and Tianwei Zhang. Real-Time Polygonal Semantic Mapping for Humanoid Robot Stair Climbing. In *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, November 2024.

R. Blickhan. The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11): 1217–1227, 1989. ISSN 0021-9290. doi: https://doi.org/10.1016/0021-9290(89)90224-8. URL https://www.sciencedirect.com/science/article/pii/0021929089902248.

H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems*. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984. ISSN 1474-6670. doi: https://doi.org/10.1016/S1474-6670(17)61205-9. URL https://www.sciencedirect.com/science/article/pii/S1474667017612059. 9th IFAC World Congress: A Bridge Between Control Science and Technology, Budapest, Hungary, 2-6 July 1984.

Paul T Boggs and Jon W Tolle. Sequential quadratic programming. *Acta numerica*, 4:1–51, 1995.

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

Boston Dynamics. Starting on the Right Foot with Reinforcement Learning. URL https://bostondynamics.com/blog/starting-on-the-right-foot-with-reinforcement-learning/.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Now Foundations and Trends, 2011. doi: 10.1561/2200000016.

Duncan Calvert, Bhavyansh Mishra, Stephen McCrory, Sylvain Bertrand, Robert Griffin, and Jerry Pratt. A Fast, Autonomous, Bipedal Walking Behavior over Rapid Regions. In *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pages 24–31, November 2022. doi: 10.1109/Humanoids53995.2022.10000120.

Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *IEEE International Symposium on System Integrations (SII)*, 2019.

Alejandro M. Castro, Frank N. Permenter, and Xuchen Han. An unconstrained convex formulation of compliant contact. *IEEE Transactions on Robotics*, 39(2):1301–1320, 2023. doi: 10.1109/TRO.2022.3209077.

Joel Chestnutt, James Kuffner, Koichi Nishiwaki, and Satoshi Kagami. Planning biped navigation strategies in complex environments. In *IEEE Int. Conf. on Humanoid Robotics (Humanoids' 03)*, 2003.

Jia-Ruei Chiu, Jean-Pierre Sleiman, Mayank Mittal, Farbod Farshidian, and Marco Hutter. A collision-free mpc for whole-body dynamic locomotion and manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4686–4693, 2022. doi: 10.1109/ICRA46639.2022.9812280.

Thomas Corbères, Carlos Mastalli, Wolfgang Merkt, Ioannis Havoutis, Maurice Fallon, Nicolas Mansard, Thomas Flayols, Sethu Vijayakumar, and Steve Tonneau. Perceptive Locomotion through Whole-Body MPC and Optimal Region Selection, May 2023.

Hongkai Dai and Russ Tedrake. Optimizing robust limit cycles for legged locomotion on unknown terrain. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 1207–1213, 2012. doi: 10.1109/CDC.2012.6425971.

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, May 2022. doi: 10.1109/ICRA46639.2022.9812247.

Robin Deits and Russ Tedrake. Footstep planning on uneven terrain with mixed-integer convex optimization. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 279–286, November 2014. doi: 10.1109/HUMANOIDS.2014.7041373.

Moritz Diehl, Hans Georg Bock, and Johannes P. Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736, 2005. doi: 10.1137/S0363012902400713. URL https://doi.org/10.1137/S0363012902400713.

Yanran Ding, Chuanzheng Li, and Hae-Won Park. Kinodynamic Motion Planning for Multi-Legged Robot Jumping via Mixed-Integer Convex Program. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3998–4005, October 2020. doi: 10.1109/IROS45743.2020.9341572.

Oluwami Dosunmu-Ogunbi, Aayushi Shrivastava, Grant Gibson, and Jessy W. Grizzle. Stair Climbing using the Angular Momentum Linear Inverted Pendulum Model and Model Predictive Control, July 2023.

Helei Duan, Ashish Malik, Jeremy Dao, Aseem Saxena, Kevin Green, Jonah Siekmann, Alan Fern, and Jonathan Hurst. Sim-to-Real Learning of Footstep-Constrained Bipedal Dynamic Walking. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10428–10434, May 2022. doi: 10.1109/ICRA46639.2022.9812015.

Helei Duan, Bikram Pandit, Mohitvishnu S Gadde, Bart Jaap van Marum, Jeremy Dao, Chanho Kim, and Alan Fern. Learning vision-based bipedal locomotion for challenging terrain. *arXiv preprint arXiv:2309.14594*, 2023.

Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Three-dimensional bipedal walking control using divergent component of motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2600–2607, 2013. doi: 10.1109/IROS.2013.6696723.

Michael Ernst, Harmut Geyer, and Reinhard Blickhan. *Spring-Legged Locomotion on Uneven Ground: A Control Approach to Keep the Running Speed Constant*, pages 639–644. aug

2009. doi: 10.1142/9789814291279_0078. URL https://www.worldscientific.com/doi/abs/10.1142/9789814291279_0078.

Maurice Fallon and Matt Antone. Plane Seg – Robustly and Efficiently Extracting Contact Regions from Depth Data, 2019. URL https://github.com/ori-drs/plane_seg.

Maurice F. Fallon, Pat Marion, Robin Deits, Thomas Whelan, Matthew Antone, John McDonald, and Russ Tedrake. Continuous humanoid locomotion over uneven terrain using stereo fusion. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 881–888, November 2015. doi: 10.1109/HUMANOIDS.2015.7363465.

Péter Fankhauser, Michael Bloesch, and Marco Hutter. Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization. *IEEE Robotics and Automation Letters*, 3(4):3019–3026, October 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2849506.

Nolan Fey, Robert J. Frei, and Patrick M. Wensing. 3D Hopping in Discontinuous Terrain Using Impulse Planning with Mixed-Integer Strategies. *IEEE Robotics and Automation Letters*, pages 1–8, 2024. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2024.3397528.

R. J. Full and D. E. Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of Experimental Biology*, 202(23):3325–3332, 12 1999. ISSN 0022-0949. doi: 10.1242/jeb.202.23.3325. URL https://doi.org/10.1242/jeb.202.23.3325.

Mariano Garcia, Anindya Chatterjee, Andy Ruina, and Michael Coleman. The Simplest Walking Model: Stability, Complexity, and Scaling. *Journal of Biomechanical Engineering*, 120 (2):281–288, April 1998. ISSN 0148-0731. doi: 10.1115/1.2798313. URL https://doi.org/10.1115/1.2798313. _eprint: https://asmedigitalcollection.asme.org/biomechanical/article-pdf/120/2/281/5766768/281_1.pdf.

Carlos E. García, David M. Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(89)90002-2. URL https://www.sciencedirect.com/science/article/pii/0005109889900022.

Hartmut Geyer, Andre Seyfarth, and Reinhard Blickhan. Compliant leg behaviour explains basic dynamics of walking and running. *Proceedings of the Royal Society B: Biological Sciences*, 273 (1603):2861–2867, 2006. doi: 10.1098/rspb.2006.3637. URL https://royalsocietypublishing.org/doi/abs/10.1098/rspb.2006.3637.

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, October 2022. doi: 10.1109/IROS47612.2022.9981969.

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, May 2021. doi: 10.1109/ICRA48506.2021.9560821.

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, September 2018.

Ambarish Goswami. Postural stability of biped robots and the foot-rotation indicator (fri) point. *The International Journal of Robotics Research*, 18(6):523–533, 1999. doi: 10.1177/02783649922066376. URL https://doi.org/10.1177/02783649922066376.

Ruben Grandia, Fabian Jenelten, Shaohui Yang, Farbod Farshidian, and Marco Hutter. Perceptive Locomotion through Nonlinear Model Predictive Control, August 2022.

Robert Griffin, James Foster, Stefan Pasano, Brandon Shrewsbury, and Sylvain Bertrand. Reachability Aware Capture Regions with Time Adjustment and Cross-Over for Step Recovery. In *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pages 1–8, December 2023. doi: 10.1109/Humanoids57100.2023.10375180.

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)*, pages 9–16, 2019. doi: 10.1109/Humanoids43949.2019.9035046.

Zhaoyuan Gu, Nathan Boyd, and Ye Zhao. Reactive locomotion decision-making and robust motion planning for real-time perturbation recovery. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1896–1902, 2022. doi: 10.1109/ICRA46639.2022.9812068.

Zhaoyuan Gu, Yuntian Zhao, Yipu Chen, Rongming Guo, Jennifer K. Leestma, Gregory S. Sawicki, and Ye Zhao. Robust-Locomotion-by-Logic: Perturbation-Resilient Bipedal Locomotion via Signal Temporal Logic Guided Model Predictive Control, March 2024.

Zhaoyuan Gu, Junheng Li, Wenlan Shen, Wenhao Yu, Zhaoming Xie, Stephen McCrory, Xianyi Cheng, Abdulaziz Shamsah, Robert Griffin, C. Karen Liu, Abderrahmane Kheddar, Xue Bin Peng, Yuke Zhu, Guanya Shi, Quan Nguyen, Gordon Cheng, Huijun Gao, and Ye Zhao. Humanoid locomotion and manipulation: Current progress and challenges in control, planning, and learning, 2025. URL https://arxiv.org/abs/2501.02116.

C.R. Hargraves and S.W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987. doi: 10.2514/3.20223. URL https://doi.org/10.2514/3.20223.

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, March 2020. ISSN 0278-3649. doi: 10.1177/0278364919894385.

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)*, pages 1447–1454, 2016. doi: 10.1109/ICRA.2016.7487279.

Armin Hornung and Maren Bennewitz. Adaptive level-of-detail planning for efficient humanoid navigation. In *2012 IEEE International Conference on Robotics and Automation*, pages 997–1002, 2012. doi: 10.1109/ICRA.2012.6224898.

Taylor Howell, Nimrod Gileadi, Saran Tunyasuvunakool, Kevin Zakka, Tom Erez, and Yuval Tassa. Predictive Sampling: Real-time Behaviour Synthesis with MuJoCo, dec 2022. URL https://arxiv.org/abs/2212.00541.

Taylor A. Howell, Brian E. Jackson, and Zachary Manchester. Altro: A fast solver for constrained trajectory optimization. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7674–7679, 2019. doi: 10.1109/IROS40897.2019.8967788.

Albert S. Huang, Edwin Olson, and David C. Moore. LCM: Lightweight Communications and Marshalling. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4057–4062, October 2010. doi: 10.1109/IROS.2010.5649358.

Marco Hutter, C. David Remy, Mark A. Höpflinger, and Roland Siegwart. Slip running with an articulated robotic leg. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4934–4939, 2010. doi: 10.1109/IROS.2010.5651461.

David H. Jacobson and David Q. Mayne. *Differential Dynamic Programming*. Elsevier, New York, 1970.

Fabian Jenelten, Takahiro Miki, Aravind E Vijayan, Marko Bjelonic, and Marco Hutter. Perceptive locomotion in rough terrain – online foothold optimization. *IEEE Robotics and Automation Letters*, 5(4):5370–5376, 2020. doi: 10.1109/LRA.2020.3007427.

Fabian Jenelten, Ruben Grandia, Farbod Farshidian, and Marco Hutter. TAMOLS: Terrain-Aware Motion Optimization for Legged Systems. *IEEE Transactions on Robotics*, 38(6):3395–3413, December 2022. ISSN 1941-0468. doi: 10.1109/TRO.2022.3186804.

Fabian Jenelten, Junzhe He, Farbod Farshidian, and Marco Hutter. Dtc: Deep tracking control. *Science Robotics*, 9(86):eadh5401, 2024. doi: 10.1126/scirobotics.adh5401. URL https://www.science.org/doi/abs/10.1126/scirobotics.adh5401.

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 (IROS)*, volume 1, pages 239–246

vol.1, October 2001. doi: 10.1109/IROS.2001.973365.

S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pages 1620–1626 vol.2, 2003. doi: 10.1109/ROBOT.2003.1241826.

Sotaro Katayama and Toshiyuki Ohtsuka. Efficient solution method based on inverse dynamics for optimal control problems of rigid body systems. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2070–2076, 2021. doi: 10.1109/ICRA48506.2021.9561109.

Majid Khadiv, Alexander Herzog, S. Ali A. Moosavian, and Ludovic Righetti. Walking Control Based on Step Timing Adaptation, March 2020.

O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987. doi: 10.1109/JRA.1987.1087068.

Charles Khazoom, Seungwoo Hong, Matthew Chignoli, Elijah Stanger-Jones, and Sangbae Kim. Tailoring solution accuracy for fast whole-body model predictive control of legged robots. *IEEE Robotics and Automation Letters*, 9(12):11074–11081, 2024. doi: 10.1109/LRA.2024.3455907.

Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *arXiv preprint arXiv:1909.06586*, 2019.

Gijeong Kim, Donghun Kang, Joon-Ha Kim, Seungwoo Hong, and Hae-Won Park. Contact-implicit model predictive control: Controlling diverse quadruped motions without pre-planned contact modes or trajectories. *The International Journal of Robotics Research*, 0(0), 2024. doi: 10.1177/02783649241273645.

J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Online footstep planning for humanoid robots. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 1, pages 932–937 vol.1, September 2003. doi: 10.1109/ROBOT.2003.1241712.

Scott Kuindersma, Frank Permenter, and Russ Tedrake. An efficiently solvable quadratic program for stabilizing dynamic locomotion. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2589–2594, 2014. doi: 10.1109/ICRA.2014.6907230.

Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40 (3):429–455, March 2016. ISSN 1573-7527. doi: 10.1007/s10514-015-9479-3. URL https:

//doi.org/10.1007/s10514-015-9479-3.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. 2021.

Simon Le Cleac'h, Mac Schwager, Zachary Manchester, Vikas Sindhwani, Pete Florence, and Sumeet Singh. Single-level differentiable contact simulation. *IEEE Robotics and Automation Letters*, 8 (7):4012–4019, 2023. doi: 10.1109/LRA.2023.3268824.

He Li and Patrick M. Wensing. Cafe-Mpc: A Cascaded-Fidelity Model Predictive Control Framework with Tuning-Free Whole-Body Control, March 2024.

Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *First International Conference on Informatics in Control, Automation and Robotics*, volume 2, pages 222–229. SciTePress, 2004.

Jyh-Ming Lien and Nancy M. Amato. Approximate convex decomposition of polygons. *Computational Geometry*, 35(1):100–123, August 2006. ISSN 0925-7721. doi: 10.1016/j.comgeo.2005.10.005.

J. Y. S. Luh, M. W. Walker, and R. P. C. Paul. On-Line Computational Scheme for Mechanical Manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 102 (2):69–76, June 1980. ISSN 0022-0434. doi: 10.1115/1.3149599. URL https://doi.org/10.1115/1.3149599. _eprint: https://asmedigitalcollection.asme.org/dynamicsystems/article-pdf/102/2/69/5696694/69_1.pdf.

Tobia Marcucci, Jack Umenberger, Pablo Parrilo, and Russ Tedrake. Shortest paths in graphs of convex sets. *SIAM Journal on Optimization*, 34(1):507–532, 2024.

Carlos Mastalli, Saroj Prasad Chhatoi, Thomas Corbéres, Steve Tonneau, and Sethu Vijayakumar. Inverse-dynamics mpc via nullspace resolution. *IEEE Transactions on Robotics*, 39(4):3222–3241, 2023. doi: 10.1109/TRO.2023.3262186.

Stephen McCrory, Bhavyansh Mishra, Robert Griffin, Jerry Pratt, and Hakki Erhan Sevil. Bipedal navigation planning over rough terrain using traversability models. In *SoutheastCon 2023*, pages 89–95, 2023. doi: 10.1109/SoutheastCon51012.2023.10115107.

Tad McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9(2):62–82, 1990. doi: 10.1177/027836499000900206. URL https://doi.org/10.1177/027836499000900206.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, January 2022a. doi: 10.1126/scirobotics.abk2822.

Takahiro Miki, Lorenz Wellhausen, Ruben Grandia, Fabian Jenelten, Timon Homberger, and Marco

Hutter. Elevation Mapping for Locomotion and Navigation using GPU, April 2022b.

Bhavyansh Mishra, Duncan Calvert, Sylvain Bertrand, Stephen McCrory, Robert Griffin, and Hakki Erhan Sevil. GPU-Accelerated Rapid Planar Region Extraction for Dynamic Behaviors on Legged Robots. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8493–8499, September 2021. doi: 10.1109/IROS51168.2021.9636009. URL https://ieeexplore.ieee.org/document/9636009/?arnumber=9636009. ISSN: 2153-0866.

Khai Nguyen, Sam Schoedel, Anoushka Alavilli, Brian Plancher, and Zachary Manchester. Tinympc: Model-predictive control on resource-constrained microcontrollers. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7, 2024. doi: 10.1109/ICRA57147.2024.10610987.

Quan Nguyen, Ayonga Hereid, Jessy W. Grizzle, Aaron D. Ames, and Koushil Sreenath. 3D dynamic walking on stepping stones with control barrier functions. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 827–834, Las Vegas, NV, USA, December 2016. IEEE. ISBN 978-1-5090-1837-6. doi: 10.1109/CDC.2016.7798370.

Quan Nguyen, Xingye Da, J. W. Grizzle, and Koushil Sreenath. *Dynamic Walking on Stepping Stones with Gait Library and Control Barrier Functions*, pages 384–399. Springer International Publishing, Cham, 2020. ISBN 978-3-030-43089-4. doi: 10.1007/978-3-030-43089-4_25. URL https://doi.org/10.1007/978-3-030-43089-4_25.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, 2 edition, 2006. ISBN 978-0-387-30303-1.

Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall PTR, USA, 4th edition, 2001. ISBN 0130609072.

Giulia Piovan and Katie Byl. Approximation and control of the slip model dynamics via partial feedback linearization and two-element leg actuation strategy. *IEEE Transactions on Robotics*, 32(2):399–412, 2016. doi: 10.1109/TRO.2016.2529649.

Michael Posa, Scott Kuindersma, and Russ Tedrake. Optimization and stabilization of trajectories for constrained dynamical systems. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1366–1373, 2016. doi: 10.1109/ICRA.2016.7487270.

Matthew J. Powell and Aaron D. Ames. Mechanics-based control of underactuated 3D robotic walking: Dynamic gait generation under torque constraints. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 555–560, Daejeon, South Korea, October 2016. IEEE. ISBN 978-1-5090-3762-9. doi: 10.1109/IROS.2016.7759108.

Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. Capture point: A step toward humanoid push recovery. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 200–207, 2006. doi: 10.1109/ICHR.2006.321385.

Marc H Raibert and H Benjamin Brown. Dynamically Stable Legged Locomotion. 1983.

Marc H. Raibert, H. Benjamin Brown, and Michael Chepponis. Experiments in Balance with a 3D One-Legged Hopping Machine. *The International Journal of Robotics Research*, 3(2):75–92, June 1984. ISSN 0278-3649. doi: 10.1177/027836498400300207.

Jenna Reher and Aaron D. Ames. Inverse Dynamics Control of Compliant Hybrid Zero Dynamic Walking. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2040–2047, May 2021. doi: 10.1109/ICRA48506.2021.9560906.

Jenna Reher, Eric A. Cousineau, Ayonga Hereid, Christian M. Hubicki, and Aaron D. Ames. Realizing dynamic and efficient bipedal locomotion on the humanoid robot durus. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1794–1801, 2016. doi: 10.1109/ICRA.2016.7487325.

Fanny Risbourg, Thomas Corbères, Pierre-Alexandre Léziart, Thomas Flayols, Nicolas Mansard, and Steve Tonneau. Real-time Footstep Planning and Control of the Solo Quadruped Robot in 3D Environments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12950–12956, October 2022. doi: 10.1109/IROS47612.2022.9981539.

Giulio Romualdi, Stefano Dafarra, Giuseppe L'Erario, Ines Sorrentino, Silvio Traversaro, and Daniele Pucci. Online non-linear centroidal mpc for humanoid robot locomotion with step adjustment. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10412–10419, 2022. doi: 10.1109/ICRA46639.2022.9811670.

Russ Tedrake and the Drake Development Team. Drake: Model-Based Design and Verification for Robotics, 2019.

Ruwen Schnabel, Roland Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26, 2007. URL https://api.semanticscholar.org/CorpusID:12349413.

Jaehyun Shim, Carlos Mastalli, Thomas Corbères, Steve Tonneau, Vladimir Ivan, and Sethu Vijayakumar. Topology-Based MPC for Automatic Footstep Placement and Contact Surface Selection. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12226–12232, London, United Kingdom, May 2023. IEEE. ISBN 9798350323658. doi: 10.1109/ICRA48891.2023.10160333.

Jonah Siekmann, Kevin Green, John Warila, Alan Fern, and Jonathan Hurst. Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning. In *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, July 2021. ISBN 978-0-9923747-7-8. doi: 10.15607/RSS.2021.XVII.061.

Jean-Pierre Sleiman, Farbod Farshidian, Maria Vittoria Minniti, and Marco Hutter. A unified mpc framework for whole-body dynamic locomotion and manipulation. *IEEE Robotics and Automation Letters*, 6(3):4688–4695, 2021. doi: 10.1109/LRA.2021.3068908.

Daeun Song, Pierre Fernbach, Thomas Flayols, Andrea Del Prete, Nicolas Mansard, Steve Tonneau, and Young J. Kim. Solving Footstep Planning as a Feasibility Problem Using L1-Norm Minimization. *IEEE Robotics and Automation Letters*, 6(3):5961–5968, July 2021. ISSN 2377-3766. doi: 10.1109/LRA.2021.3088797.

B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi: 10.1007/s12532-020-00179-2. URL https://doi.org/10.1007/s12532-020-00179-2.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Reza Takapoui, Nicholas Moehle, Stephen Boyd, and Alberto Bemporad. A simple effective heuristic for embedded mixed-integer quadratic programming. In *2016 American Control Conference (ACC)*, pages 5619–5625, 2016. doi: 10.1109/ACC.2016.7526551.

Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175, 2014. doi: 10.1109/ICRA.2014.6907001.

Russ Tedrake. *Underactuated Robotics*. 2023. URL https://underactuated.csail.mit.edu.

Russ Tedrake, Scott Kuindersma, Robin Deits, and Kanako Miura. A closed-form solution for real-time zmp gait generation and feedback stabilization. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 936–940, 2015. doi: 10.1109/HUMANOIDS.2015.7363473.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Steve Tonneau, Daeun Song, Pierre Fernbach, Nicolas Mansard, Michel Taix, and Andrea Del Prete. SL1M: Sparse L1-norm Minimization for contact planning on uneven terrain. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6604–6610, Paris, France, May 2020. IEEE. ISBN 978-1-72817-395-5. doi: 10.1109/ICRA40945.2020.9197371.

Patrick M. Wensing and David E. Orin. Generation of dynamic humanoid behaviors through task-space control with conic optimization. In *2013 IEEE International Conference on Robotics and Automation*, pages 3103–3109, May 2013a. doi: 10.1109/ICRA.2013.6631008.

Patrick M. Wensing and David E. Orin. High-speed humanoid running through control with a 3d-slip model. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5134–5140, 2013b. doi: 10.1109/IROS.2013.6697099.

Patrick M Wensing, Michael Posa, Yue Hu, Adrien Escande, Nicolas Mansard, and Andrea Del Prete. Optimization-based control for dynamic legged robots. *IEEE Transactions on Robotics (TRO)*, October 2023. doi: 10.1109/TRO.2023.3324580. URL https://ieeexplore.ieee.org/document/10286076.

Eric R. Westervelt, Jessy W. Grizzle, and Daniel E. Koditschek. Hybrid zero dynamics of planar biped walkers. *IEEE transactions on automatic control*, 48(1):42–56, 2003.

Eric R. Westervelt, Jessy W. Grizzle, Christine Chevallereau, Jun Ho Choi, and Benjamin Morris. *Feedback Control of Dynamic Bipedal Robot Locomotion*. CRC Press, 1 edition, 2007. doi: 10.1201/9781420053739.

David Wisth, Marco Camurri, and Maurice Fallon. Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots. *IEEE Transactions on Robotics*, 39(1):309–326, 2023. doi: 10.1109/TRO.2022.3193788.

Zhaoyang Xiang, Victor Paredes, and Ayonga Hereid. Adaptive Step Duration for Precise Foot Placement: Achieving Robust Bipedal Locomotion on Terrains with Restricted Footholds, March 2024.

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, August 2022. ISSN 1941-0468. doi: 10.1109/TRO.2022.3150219.

Haoru Xue, Chaoyi Pan, Zeji Yi, Guannan Qu, and Guanya Shi. Full-order sampling-based mpc for torque-level locomotion control via diffusion-style annealing, 2024. URL https://arxiv.org/abs/2409.15610.

Will Yang and Michael Posa. Dynamic on-palm manipulation via controlled sliding. In *Robotics: Science and Systems (RSS)*, July 2024. URL https://roboticsconference.org/program/papers/12/.

William Yang and Michael Posa. Impact-invariant control: Maximizing control authority during impacts. *arXiv preprint arXiv:2303.00817*, 2023.

Shangqun Yu, Nisal Perera, Daniel Marew, and Donghyun Kim. Learning generic and dynamic locomotion of humanoids across discrete terrains. *arXiv preprint arXiv:2405.17227*, 2024.

Ziyi Zhou, Qian Meng, Hadas Kress-Gazit, and Ye Zhao. Physically-feasible reactive synthesis for terrain-adaptive locomotion via trajectory optimization and symbolic repair, 2025. URL https://arxiv.org/abs/2503.03071.