Robust and Economical Bipedal Locomotion

by

Nils Smit-Anseeuw

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in the University of Michigan
2019

Doctoral Committee:

Professor C. David Remy, Co-Chair
Professor Ram Vasudevan, Co-Chair
Professor Jessy Grizzle
Professor Necmiye Ozay

Nils Smit-Anseeuw
nilssmit@umich.edu
ORCID iD: 0000-0002-6848-7308

To Ann Arbor, and the people there that gave me a home.

ACKNOWLEDGEMENTS

This dissertation was made possible by the love, support, and guidance of many people. The last five years have been an overwhelming period of growth and exploration, and I was fortunate to have friends and family at my side every step of the way.

First, I would like to thank my advisors and colleagues: David, who convinced me to make the jump to Ann Arbor and taught me the art of a good story; Ram, who continually expanded my horizons and never stopped raising the bar; the RAMlab, who were the first to make me feel at home; and the ROAHM lab, who showed me how a lab can become a family.

Next, I am grateful for my amazing family: Mom, my number one fan who was never anything but motivating, encouraging and supportive; Dad, who showed me how to keep improving and challenging myself even as the going got tough; Fleur, the big sister I never stopped looking up to; Aksel, who sets the bar for compassionately but relentlessly pursuing ideals; and Esmee, a constant inspiration in her traverse of our shared hurdles.

I want to thank some of the close friends in Ann Arbor that I've met along the way: the PC crew, who gave me my first home away from home, were there for me when I needed it most, and turned my life around in a very real sense; Kurt, whose fishing trips were the highlight of my summer, despite never catching many fish; Alexander, who opened my eyes to good coffee, effective outreach, and bold entrepreneurship; Tori, whose passion for her work inspired me to find the same in mine; and my biking buddies, who introduced me to an incredible hobby and escape.

Finally, I want to thank Abby, who broadened my perspective on life, kept me going while searching for the light at the end of the tunnel, and introduced me to Nova (who added a few keystrokes to this dissertation).

Thanks to these people (and many more), I found a home in Ann Arbor. Without that home, none of this would have been possible.

PREFACE

Chapters II, III, and V are first-author publications that have each been assembled as separate manuscripts. Chapter II was published in Robotics and Automation Letters 2017 Volume 2 and presented at ICRA 2017 [88]. Chapter III was published and presented at IROS 2017 [89]. Chapter V was accepted for publication in Robotics and Automation Letters 2019 and will be presented at IROS 2019 [90]. These chapters are presented here in their published form; given this, there may be some repetition of material.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

PREFACE iv

LIST OF FIGURES viii

ABSTRACT xvi

CHAPTER

I. Introduction 1

1.1 Motivation 1
1.2 State of the Art 3
1.3 Contributions 6

II. The Energetic Benefit of Robotic Gait Selection: A Case Study on the Robot RAM*one* 8

2.1 Introduction 8
2.2 Constructing Optimal Motions 10
2.2.1 Model 10
2.2.2 Optimization 13
2.3 Optimal Motions and Gaits 15
2.3.1 Ballistic Walking at Speeds Below 1.04 m/s 17
2.3.2 Spring-Mass Running at Speeds Above 1.04 m/s . . . 20
2.3.3 Walk to Run Transition 21
2.3.4 Influence of the Knee Direction 23
2.4 Discussion & Conclusion 25

III. RAM*one*: A Planar Biped for Studying the Energetics of Gait 29

3.1 Introduction 29

3.2 Methods . . . 31
3.2.1 RAM*one* Hardware . . . 31
3.2.2 Controller . . . 33
3.2.3 Simulation . . . 36
3.3 Results . . . 37
3.3.1 Walking Controller . . . 37
3.3.2 Cost of Transport . . . 40
3.4 Discussion & Conclusion . . . 42

IV. Safety as a Constraint: Viability-based Control of Hybrid Systems . . . 44

4.1 Introduction . . . 44
4.2 Problem Statement . . . 46
4.2.1 Dynamics . . . 46
4.2.2 Safety . . . 48
4.2.3 Semi-Autonomous Safe Control . . . 49
4.2.4 Goal . . . 49
4.3 Numerical Formulation . . . 50
4.3.1 Constraints . . . 50
4.3.2 Objective Function . . . 51
4.3.3 Numerical Optimization Problem . . . 52
4.4 Computational Implementation . . . 53
4.4.1 Function Representation with Polynomials . . . 53
4.4.2 Sums-of-Squares . . . 53
4.4.3 Sufficient Invariance Conditions . . . 54
4.4.4 Bilinear Sums-of-Squares Problem . . . 55
4.4.5 Alternation . . . 55
4.5 Guaranteed Safe Semi-autonomous Controller . . . 58
4.6 Results . . . 58
4.6.1 Dubins Car . . . 59
4.6.2 Compass Gait Walker . . . 61
4.7 Conclusion . . . 62

V. Walking with Confidence: Safety Regulation for Full Order Biped Models . . . 65

5.1 Introduction . . . 65
5.2 Problem Setup . . . 69
5.2.1 Robot Model . . . 69
5.2.2 Safety . . . 70
5.2.3 Goal . . . 71
5.3 Controlled Hybrid Zero Dynamics Manifold . . . 71
5.3.1 Shaping Parameters . . . 71
5.3.2 Constructing the Manifold . . . 72

5.3.3 Safety on the Manifold . . . 74
5.4 Hybrid Control Invariant Set . . . 75
5.4.1 Polynomial Representation . . . 75
5.4.2 Optimization Formulation . . . 76
5.4.3 Guaranteed Safe Semi-autonomous Controller . . . 79
5.5 Results . . . 80
5.6 Conclusion . . . 82

VI. Stepping Onto Rough Terrain: Reachability-based Gait Selection . . . 85

6.1 Introduction . . . 85
6.2 Reachability-based Gait Selection . . . 87
6.2.1 Standing Controller and Region of Attraction . . . 87
6.2.2 Mid-stance Configuration . . . 87
6.2.3 Stepping Controller and Forward Reachable Set . . . 89
6.2.4 Stopping Controller and Backwards Reachable Set . . . 91
6.2.5 Gait Selector . . . 92
6.3 Discussion and Limitations . . . 92

VII. Conclusions and Future Directions . . . 93

7.1 Discussion of Contributions . . . 94
7.2 Future Directions . . . 95
7.2.1 Safety in 3 Dimensions . . . 95
7.2.2 From Model to Reality . . . 96
7.2.3 Guaranteed Safe Online Learning . . . 96
7.3 Concluding Remarks . . . 97

APPENDIX . . . 98
A.1 Expanded Hybrid Invariance Condition . . . 99
A.2 Hybrid Invariant Targets . . . 101

BIBLIOGRAPHY . . . 104

LIST OF FIGURES

Figure

1.1 The first two aims of this proposal investigate properties of bipedal gait for the robot RAM*one*. This robot was built to explore the exploitation of natural dynamics in legged locomotion. It is based on the Scarl*ETH* leg design [45], driven by series elastic actuators, and mounted on a planarizer which restricts its motion to the sagittal plane [33]. The robot is represented as a detailed five link planar model. The model uses a floating base description with rigid rolling contacts, it encodes the actuator dynamics with non-linear springs, and accounts for dry friction and viscous damping in the joints. . . 4

2.1 Cost of transport is shown as a function of speed given in $^{\mathrm{m}}/_{\mathrm{s}}$ and as a non-dimensional Froude number: $F = \frac{v^2}{g\ell}$ (where $\ell = 0.47\,\mathrm{m}$ is the extended leg length of the robot). At all speeds, the walking sequences with an extended double stance phase were suboptimal compared to those with instantaneous support transfer. As such, the remainder of the analysis focuses on walking sequences with zero double support phase. At low speeds, all four remaining cases had a similar CoT. Walking footfall sequences had a slight energetic advantage. For speeds below $0.88\,^{\mathrm{m}}/_{\mathrm{s}}$, the walking sequences with the knees pointing forward had the lowest CoT, for speeds between $0.88\,^{\mathrm{m}}/_{\mathrm{s}}$ and $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$ it became optimal for knees to point backwards. Motions with a running footfall sequence were slightly more energetically expensive below $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$. However, their CoT was only up to 0.036 higher. At high speeds above $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$, the CoT of the four gaits diverged. Moving with a running sequence and with the knees pointing backwards became by far the most efficient mode of locomotion. Using a walking sequence increased the CoT by up to 0.77 (259 %) at $2.04\,^{\mathrm{m}}/_{\mathrm{s}}$. Having the knees point forward instead of backward increased the CoT of the running sequence by up to 0.52 (148 %) at $2.68\,^{\mathrm{m}}/_{\mathrm{s}}$. . . 16

2.2 Trajectory stills show the optimal motions at a speed of 0.9 m/s. This speed is below the walk to run transition and the gaits share similar properties that are indicative of ballistic walking. In particular, the knees are extended, the main body is pitched forward, and the center of mass moves in an upwards arc. The running sequence with knees pointing backwards is an exception to this pattern and shows characteristics more reminiscent of Groucho running. 18
2.3 Center of mass kinetic, gravitational, and elastic energy flow for all motions at 0.9 m/s. Both walking sequence motions and the knees forward running sequence motion exhibit an out-of-phase relationship between kinetic and potential energy. This is prototypical of ballistic walking [61]. The knees backwards running sequence gait exhibits Groucho running [60], characterized by a constant potential energy and an out of phase exchange between kinetic and elastic energy. . . 19
2.4 Ground reaction force magnitudes at 0.9 m/s. A distinct pushoff force is observed near the end of each single stance phase for the knees forward walking sequence. This pushoff force is conspicuously absent in the other three motions. 19
2.5 Flight phase durations of the running sequences. The optimal flight phase duration is near zero for speeds below the walk to run transition and jumps up sharply at speeds above this point, indicating a sudden and discrete change in gait. 21
2.6 Trajectory stills of the optimal running sequence motions at 1.2 m/s. In contrast to the same footfall sequence at lower speeds (Fig. 2.2), the motions now exhibit clear properties of spring-mass running. This includes a pronounced knee bend and a downwards arc of the center of mass. 22
2.7 Center of mass kinetic, gravitational, and elastic energy flow for running sequence motions at 1.2 m/s. The left and right contact phases of the motion are indicated with a shaded background. Both motions exhibit the in-phase kinetic and gravitational energy oscillation with out of phase elastic energy that characterizes spring mass running [12] 22
2.8 Total ground reaction force at each foot for running sequence motions at 1.2 m/s. Both knees forward and backwards motions exhibit a sharp initial force at contact followed by a single hump. 23

2.9 Comparison of loss contributions (normalized by $mg\Delta x$) between walking and running sequences . Results are shown across a range of speeds and for motions with the knees pointing backwards. All components of the walking sequence motion increase smoothly across speed, with electrical losses dominating. In contrast, the individual costs of the running sequence motion change discretely across the walk to run transition. Below the transition, the costs of the running sequence motion roughly follow those of the walking sequence motion. Above the transition, the electrical losses sharply decrease and damping losses dominate the cost. This sharp jump is a result of the transition from *Groucho running* to spring mass running. . . 24

2.10 The different knee directions in the running sequence motion lead to different robot configurations at liftoff (shown in (a) for a velocity of $1.2\,{}^{\mathrm{m}}/_{\mathrm{s}}$). As a consequence, the joint velocities in the swing leg (shown in (b)) differ greatly between the two cases. These joint velocities are the sum of the motor and spring velocities. A significant deflection in the knee springs decouples the joint from the motor. Joint and motor velocities are thus distinctly different, and the knee motor is already moving to retract the leg while the knee joint extends for liftoff. The same does not hold for the hip joint, where joint and motor motion must be more similar and the motor motion can only be reversed after lift-off. Because of this, running with the knees backwards is at an advantage, since it requires a much smaller hip velocity. [Note that all velocities are defined to be positive when counter clockwise. A positive knee velocity thus indicates knee extension when the knees are forward, and knee flexion when the knees are backwards.] . . . 26

3.1 **(a)** Specification of the phase parameter θ. Between -1 and 0, and between 0 and 1, θ is a linear function of the horizontal distance from the forward-most stance foot to the main body. Note that if the gait is periodic with step length l_{step}, the duration of the double stance phase x_{DS} is determined by $x_{DS} = l_{step} - x_{SS}$. The parameter Δ_x (negative as shown in the figure) is used to control the average walking speed of the robot.
(b) Gait transition sequence. Beginning in double stance, θ becomes less negative as the body moves forwards, triggering the liftoff phase when it crosses 0. In this phase, the hip motor is held fixed and the knee motor retracts until contact is no longer sensed in the swing foot. Now in single stance, θ continues to increase from 0 as the body moves forward, until touchdown is initiated when θ crosses 1. Here the swing foot is held at fixed distance in front of the body until contact is sensed. θ is then reset to -1 and we enter double stance. . 35

3.2 Stills of the robot across stride percentage in both hardware and simulation for a desired body height of 0.54 m and desired walking speed of $0.2\,{}^{\mathrm{m}}/_{\mathrm{s}}$. 37

3.3 Tracking of controller objectives in simulation and hardware across stride percentage for a desired body height of 0.54 m and desired speed of 0.2 m/s. Plotted values are averaged over 30 strides, and one standard deviation is shaded in grey. The color of each line corresponds to the current stance configuration of the robot. In **(a)** we see that body height is maintained to within 5 mm of the desired value, with a slight rise upon entering double stance that occurs in both simulation and hardware. The horizontal body speed in **(b)** varies more significantly, as it is not controlled directly. The pitch in **(c)** is controlled to within 0.04 rad in hardware, with similar variation in simulation, however the phase and timing of the oscillations are different between the two. In **(d)** and **(e)**, we show the x and y coordinates of each foot with respect to the ground. During swing phase, the desired trajectory of each foot is shown as a thick dashed line. 38

3.4 Trace of the foot trajectories with respect to the ground. We see significant deflection of the foot during stance phase (represented by negative y position). Also note that each foot slips slightly along the ground before liftoff in simulation, but not in hardware. 39

3.5 Cost of transport and nondimensionalized copper losses in the hips and knees across desired body height in hardware and simulation. A minimum in the CoT was observed in both simulation and hardware at a desired body heights of 0.56 m. Below this height, the CoT decreased linearly with increasing body height, with a slope of -2.835 1/m in hardware, and -3.08 1/m in simulation. The decrease can be attributed to the decreased knee copper losses incurred by a straighter legged gait. This is supported by the observation that on average, the knee copper losses decrease with walking height while the hip copper losses remain comparatively constant. 41

4.1 An illustration of how the objective function (4.7) approximates the volume of the viability domain. Take a set of differentiable functions v_i that satisfy the constraints of the previous section. For every point x not in the set ∂V_i, the value $v_i(x)$ is constrained only by (4.3) and $v_i(x) \leq 1, \forall x \in X_i$. This means that $v_i(x)$ can increase to a value of 1 for points inside the set (x s.t. $v_i(x) > 0$), and $v_i(x)$ increases to a value of 0 for points outside this set. As a result, each v_i approaches the indicator function over V_i, and the integral in Eq. 4.7 approaches the original objective. 52

4.2 Scaling weights between the user input u_0 and the guaranteed safe controller u. The weights satisfy $w_0 + w_s = 1$ and are used to form the semi-autonomous controller $u_m = w_0 u_0 + w_s u$. When the barrier function value is above a threshold value (i.e. $v_i(x) > v_m$), the user input is unmodified, as we are sufficiently removed from the boundary of the safe set. When $0 \leq v_i(x) \leq \frac{v_m}{2}$, the safe controller is fully active, keeping the state in the safe set. Between these regions, the controller is smoothly interpolated with a cubic spline to ensure continuity of the semi-autonomous controller. 58

4.3 Visualization of the safe set for the Dubins car showing the 0 and 0.5 level sets of the barrier function v in light green and dark green, respectively. For this example, we can compute the exact unsafe set (complement of the viability kernel), shown here in light grey. Shown in the overlay are two simulated vehicle trajectories, one (purple) uses a randomly generated input, the other (blue) combines this input with a semi-autonomous controller (threshold value $v_m = 0.5$). We see the randomly generated controller drive off the road (fail), while the semi-autonomous vehicle remains safe. The trajectories are also plotted in state space where we see that the semi-autonomous vehicle diverges only once $v = v_m$ is reached. 60

4.4 Compass gait states (right) and domains (left). The walker has two legs with mass and length parameters loosely chosen to correspond with the robot RAM*one* [89] (see supplementary code[2]). There are two bounded control inputs: a torque at the ankle and a torque between stance and swing legs. The robot has two discrete states: pre-midstance (mode 1) and post-midstance (mode 2). There is a transition defined from mode 1 to mode 2 and one defined from mode 2 to mode 1. The hybrid guard from mode 1 to mode 2 is associated with an identity reset map. The hybrid guard from mode 2 to 1 represents a touchdown event. The associated reset maps the stance leg angle to the swing leg and vice versa. 61

4.5 Safe set (shown in green) for the compass gait walker visualized as a 3D slice of the 4D state space with a stationary swing leg ($\dot{\alpha} = 0$). The yellow plane represents the touchdown guard, and the gray planes represent the edges of the failure set, i.e. the state must stay within these boundaries to remain viable. The viable set does not intersect the failure set, but does intersect the guard. Note that along the guard, increasing $\dot{\theta}$ requires an increase in α for the state to be viable. This matches the intuition that larger step lengths should be used for higher walking speeds [75]. Also note that the minimum stance leg speed is large for large stance leg angles. For lower speeds, the stance leg torque is insufficient to get the walker over mid-stance. 63

4.6 Two simulated trajectories for the compass gait model simulated using the non-Taylor-expanded dynamics. The first, in purple, uses a random control input, the other, in blue, uses our semi-autonomous controller in combination with this input (threshold value $v_m = 0.2$). In the trajectory stills we see that the raw trajectory fails by falling backwards ($\dot{\theta} < 0$), while the semi-autonomous trajectory keeps walking. Also shown is the barrier function value ($v(x)$) over time. Note the semi-autonomous controller deviates from nominal only once the threshold value (shown in light green) is reached. 64

5.1 Generating safety guarantees for a high dimensional robot (illustrated on Rabbit [18]). The state-space of the full robot is given in the top right figure, where TQ is the tangent space on Q, S is the hybrid guard representing foot touchdown, and Δ is the corresponding discrete reset map. Using feedback linearization, we restrict our states to lie on a low-dimensional manifold Z, reducing the state-space dimension to an amenable size for sums-of-squares analysis. This manifold is parameterized by the underactuated degrees of freedom of the robot θ, as well as a set of shaping parameters α. The shaping parameters can be modified in real-time by a control input, allowing for a broad range of behaviours on Z. To guarantee safety on Z we find the set of unsafe states Z^F from which the state may leave the manifold (for instance due to motor torque limits). We then use sums-of-squares tools [70] to find a control invariant set $\hat{V} \subset Z \setminus Z^F$. This control invariant set can be used to define a semi-autonomous, guaranteed safe controller for the full robot dynamics. 67

5.2 A 2D slice (along $\alpha = \dot{\alpha} = 0$) of the four-dimensional viability domain $\hat{V}$ (shown in green) for Rabbit. The border at the right corresponds to the hybrid guard $\hat{S}$ of foot touchdown, where the state is reset under the map $\hat{\Delta}$ to the left of the figure. The unsafe set Z^F is shown in red. We avoid the lower region ($\dot{\theta} < 0$) in order to conservatively prevent backwards falls. The upper region conservatively approximates the region in which the control input (5.8) violates the torque limits of the robot. By modifying the control input whenever Rabbit is at the edge of $\hat{V}$, Z^F can be avoided indefinitely. Finally, the periodic trajectory used to generate our targets q^{Fr} is shown in dashed black. Note that our viability domain is able to guarantee robot safety even for states far away from this nominal trajectory. 81

5.3 Tracking performance of the safe (5.20) and naïve (5.21) controllers following two reference trajectories under the full rabbit dynamics. The pitch angles are shown in the top left. For both references, the safe controller modifies the input before safety is at risk, while the naïve controller follows the reference even as it leads to failure. Failure for the upper trajectory corresponds to stepping backwards, and in the lower trajectory corresponds to moving too fast for the swing leg to reach its target. The bottom left figure shows desired input u_α^d and executed input for both naïive and safe tracking controllers following the second reference target. The state-dependent region of inputs that satisfy the torque constraints are shown in grey. Note that under the naïve controller, this region vanishes as the forward walking speed of the robot becomes too high. Stills from the simulation trajectories are shown on the right. The dotted line is the desired pitch, and the faded line is the nearest on-manifold state $q_0(\theta, \alpha)$. . 83

6.1 Overview of the reachability-based gait selection approach for a Rabbit model walking over rough terrain. The approach begins by constructing a standing controller that stabilizes the robot about a stationary standing position. Next, we specify a mid-stance configuration q_m and configuration velocity $\frac{\partial q_m}{\partial \theta}$. We then generate a parametrized family of first steps (using interpolated FROST trajectories) that start and end at the mid-stance configuration for a range of step-lengths (x_1), step heights (y_1), and accelerations (a). Next we construct a stopping controller (using the methods of the previous chapter) that brings the robot to a stop in one step (of step-length x_2 and step-height y_2) by actively modifying the pitch angle. Finally, we compute three set-based objects: a *region of attraction* (ROA) of the standing controller, a *forward reachable set* (FRS) of the first-step controller, and a *backwards reachable set* of the ROA using the second-step stopping controller. The FRS takes in the current mid-stance velocity ($\dot{\theta}_0$) and the discrete parameters of the first step (β_1), and returns a bound on the next mid-stance velocity ($\dot{\theta}_1$). The BRS takes in the parameters of the stopping step (β_2), and returns the set of mid-stance velocities that can safely be brought to a stop in one step. At each mid-stance event, the gait-selector searches for discrete step parameters (β_1, β_2) that can bring the robot to a stop in two steps (i.e. $FRS(\dot{\theta}_0, \beta_1) \subset BRS(\beta_2)$). If the selector succeeds, the robot takes the first step using parameters values β_1, and the selection process repeats at the next mid-stance. If the selector fails, the robot executes a stopping step using the parameter values β_2 from the previous mid-stance. Since the previous selection step ensured that the robot state is within the BRS of the stopping controller, we know that the robot can safely be brought to a stop. 88

7.1 The bipedal robot Cassie (left) and a realistic model (right). Image taken from the video *Cassie Sim Comparison* released by Agility Robotics [80]. 95

ABSTRACT

For bipedal robots to gain widespread use, significant improvements must be made in their energetic economy and robustness against falling. An increase in economy can increase their functional range, while a reduction in the rate of falling can reduce the need for human intervention. This dissertation explores novel concepts that improve these two goals in a fundamental manner. By centering on core ideas instead of direct application, these concepts are aimed at influencing a wide range of current and future legged robots.

The presented work can be broken into five major contributions. The first extends our understanding of the energetic economy of series elastic walking robots. This investigation uses trajectory optimization to find energy-miminizing periodic motions for a realistic model of the walking robot RAM*one*. The energetically optimal motions for this model are shown to closely resemble human walking at low speeds, and as the speed increases, the motions switch abruptly to those resembling human running. The second contribution explores the energetic economy of the real robot RAM*one*. Here the model used in the previous investigation is shown to closely match reality. In addition, this investigation demonstrates a concrete example of a trade-off between energetic economy and robustness. The third contribution takes a step towards addressing this trade-off by deriving a robot constraint that guarantees safety against falling. Such a constraint can be used to remove considerations of robustness while conducting future investigations into economical robot motions. The approach is demonstrated using a simple compass-gait style walking model. The fourth contribution extends this safety constraint towards higher-dimensional walking models, using a combination of hybrid zero dynamics and sums-of-squares analysis. This is demonstrated by safely modifying the pitch of a 10 dimensional Rabbit model walking over flat terrain. The final contribution pushes the safety guarantee towards a broader set of walking behaviours, including rough terrain walking.

Throughout this work, a range of models are used to reason about the economy and robustness of walking robots. These model-based methods allow control designers to move away from heuristics and tuning, and towards generalizable and reliable controllers. This is vital for walking robots to push further into the wild.

CHAPTER I

Introduction

1.1 Motivation

Society relies heavily on wheeled vehicles for transportation and mobility. From trains and cars to bicycles and scooters, wheels have greatly improved our ability to move around. Despite their wide-spread use, wheeled vehicles are relatively limited in the terrain they can traverse. In cities, wheels are restricted to clear roads and walkways; when obstacles are in the way or snow is on the ground, wheeled vehicles will often come to a stop. Off of roads and walkways, these limitations are amplified. Brush, rocks, mud, and sand can prove impassable even for designated off-road vehicles.

In remarkable contrast, many humans and legged animals have little difficulty traversing such terrain. Legged robots, like humans and animals, have the potential to access a wider variety of landscapes than their wheeled counterparts. Bipedal robots in particular will be able to access spaces that are specifically designed for people, including vehicle interiors, ladders and stairways. Until recently, bipedal robots were confined to controlled, known lab environments. However, over the last five years, they have started to leave the lab. We have seen demonstrations of bipedal robots walking through the woods [81], running and jumping over obstacles [24], and walking through sand, snow and grass [31]. There are many proposed applications that come with this increase in range. Search and rescue, last mile delivery, construction site monitoring, and security patrolling are among the many stated targets of companies such as Boston Dynamics and Agility Robotics.

So what is missing? Why are legged robots not out in the world, accessing all the space open to people? It is instructive to first look at what humans do well when locomoting. Take an athlete running a long distance trail race (such as the Leadville 100: a rugged 100 mile race through the Rocky Mountains). The first observation

from such an athlete is that despite the difficult and dangerous terrain, they can move with confidence in their safety. Since falling comes with a high risk of injury, confidence against falling is a crucial requirement for human locomotion. The second observation is that this athlete can traverse incredible distance and elevation with very little fuel. In one study, ultra marathon runners were found to consume just 4200 kJ over the course of a 100 km race [26] (this would be equivalent to a vehicle fuel economy of 1765 mpg!). We call this *energetic economy*, and measure it by the cost of transport (CoT)[1]. People are known to have a cost of transport around 0.376 when walking on flat terrain [99].

How do robots compare? Those that seem to be most robust and versatile, such as Atlas [23] and Cassie [79], use powerful actuation for high-precision control, enabling them to perform fast and strong actions to react to and cancel large perturbations. However, despite this powerful actuation, these robots still fall regularly, even on flat terrain. Powerful actuation also comes at an energetic cost. For example, Atlas has a CoT of ~ 5 [11], and ATRIAS (Cassie's predecessor) has a CoT of 1.13 [43], more than 3 times that of a human. In contrast, the robot Cornell Ranger is about as energy-economical as humans, with a CoT of 0.28 [10]. Ranger was designed and controlled specifically to walk with little power, e.g. it exploits natural dynamic motions and performs work when it is most optimal to do so ('preemptive push-off'). But Ranger is barely stable, as it falls on slopes of just two degrees and is only able to walk, but not, say, run or hop. This apparent trade-off between robustness and economy has also been observed when designing legged robot controllers, as more robust controllers have been seen to incur an energetic penalty for the robot Rabbit [83].

The goal of this thesis is to help bridge the gap between robots and people by making fundamental improvements in the robustness and economy of bipedal legged robots. I aim to explore four major questions:

1. How should a realistic robot model locomote in order to minimize energy consumption?

2. Can such economical motion translate from a model to reality?

3. Can we constrain simple legged robot models to avoid falls while allowing for flexible behaviour?

4. Can such a constraint be extended to more complex robot models?

[1]Cost of transport is a standard metric [98], defined as the energy consumed per distance traveled normalized by body weight

A central theme that underlies these questions is how different models across a range of complexity and abstraction can be used to reason about the robustness and economy of legged robots. The models used in our investigation include *simple models* that capture the general underlying principles of locomotion, *detailed models* that closely approximate the behaviour of specific robots, real robots (i.e. *physical models*) such as the planar bipedal robot RAM*one* (Fig. 1.1) which can be used to evaluate performance in experiment, and *biological models* which provide a benchmark for performance as well as an illustration of important locomotion principles. We explore the strengths and limitations of these models, and use them to explicitly construct conclusions and guarantees about bipedal locomotion while acknowledging their fundamental limitations.

Using these models, I will present the following principle findings. First, RAM*one* should use different gaits at different speeds to minimize energy consumption, and these gaits bear strong similarity to templates of animal motion found in nature. Next, straight legged walking is economical for the real robot RAM*one*, and economical motions can lie at the boundary of failure, where falls are highly likely. Third, a safety regulator that guarantees against falling can be generated for simple biped models. Finally, this safety regulator can be extended to more complex models with a wider range of behaviours.

1.2 State of the Art

Significant progress has been made towards improving the *energetic economy* and the *robustness* of legged robots. This work can be classified as improving either *hardware* or *controller* design.

When designing robot *hardware* for improving *energetic economy*, it is important to first use hardware components that minimize energy losses in the system, regardless of the specific motion or controller. These include efficient actuators (e.g. DC motors instead of hydraulics), small friction between different elements, and regenerative power. The MIT Cheetah is one robot that effectively makes use of such hardware components [86]. In addition, it is important to use hardware design principles that can be exploited by the controller for economical gaits. For example, light legs with small feet allow for swinging the legs quickly and with little effort [86]. Springs can reduce ground-impact energy losses; they can also store energy generated by negative work and release it later to help power the robot motion [2]. ATRIAS [42], Cassie, and DURUS [76] are examples of robots that follow such principles.

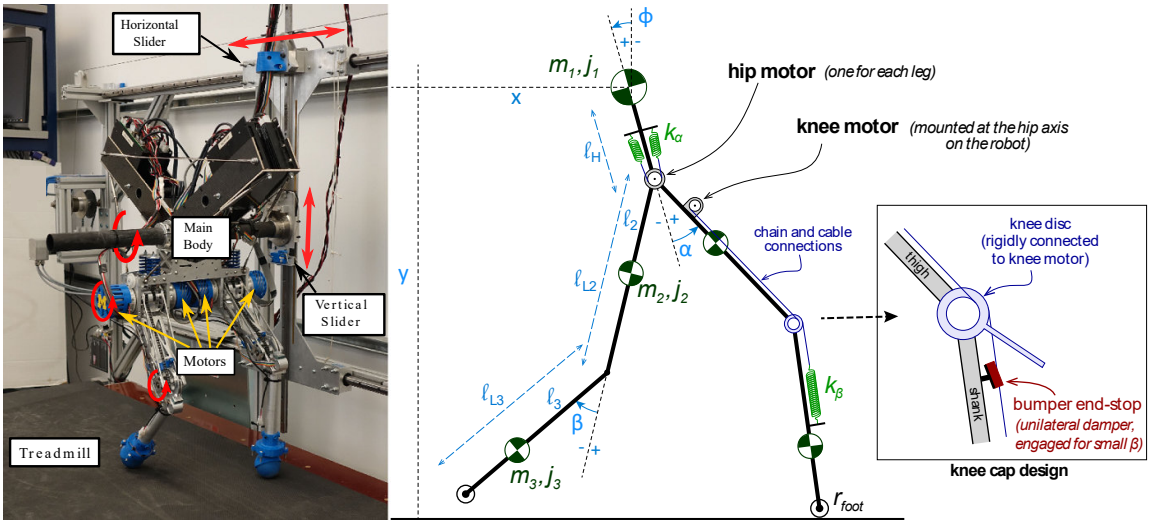

Figure 1.1: The first two aims of this proposal investigate properties of bipedal gait for the robot RAM*one*. This robot was built to explore the exploitation of natural dynamics in legged locomotion. It is based on the Scarl*ETH* leg design [45], driven by series elastic actuators, and mounted on a planarizer which restricts its motion to the sagittal plane [33]. The robot is represented as a detailed five link planar model. The model uses a floating base description with rigid rolling contacts, it encodes the actuator dynamics with non-linear springs, and accounts for dry friction and viscous damping in the joints.

Legged robot *hardware* can also be designed to improve *robustness* against falling. One such principle is that the robot should have sufficient and fast actuation in all joints in order to be able to quickly respond to disturbance. Another principle that can improve robot robustness is to have compliance in the point of contact with the ground, in order to isolate the main body motion from unexpected variation in the terrain [74, 92, 44, 43].

When designing *controllers* to improve the *energetic economy* of legged robots, one way to gain insight is to observe the characteristic properties of economical biological gait. One widespread observation is that as locomotion speed varies in many animals, the type of locomotion changes [40, 16, 3]. At low speeds, many animals exhibit an inverted pendulum style walking gait [61]. At high speeds, a spring-mass style running gait [12] is observed across a broad range of species. Transferring these observations from biology to robotics is nontrivial, as biological and artificial systems are fundamentally dissimilar. To establish a meaningful connection between nature and robotics, a number of simple models have been used to distill fundamental principles from locomotion in nature and provide templates for design and control of robots. Such models include a point mass on massless legs [94, 93], passive dynamic walkers

[59], the spring loaded inverted pendulum [12], and elastic walking models [29, 28]. The first aim of this dissertation can be seen as an extension of previous work in optimal gait for conceptual models [93, 77, 107, 108] and builds upon optimization methodologies put forward in [17, 19, 109].

A promising tool for analyzing and improving the *robustness* of legged system *controllers* is sums-of-squares (SoS) optimization [70]. This approach uses semi-definite programming to find safe sets of states and associated controllers for a broad class of nonlinear [56, 37, 50] and hybrid systems [72, 87, 62]. These safe sets can take the form of *reachable sets* (sets that can reach a known safe state) [49, 87, 56] or *invariant sets* (sets whose members can be controlled to remain in the set indefinitely) [8, 72, 71] in state space. However, the representation of each of these sets in state space severely restricts the size of the problem that can be tackled by these approaches. To accommodate this limitation, sums-of-squares analysis has been primarily applied to reduced models of walking robots: ranging from spring mass models [111], to inverted pendulum models [49, 96] and to inverted pendulum models with an offset torso mass [71]. The substantial differences between these simple models and real robots causes difficulty when applying these results to hardware.

A contrasting approach to designing robust controllers for high dimensional, underactuated robot models uses hybrid zero dynamics (HZD) [104]. In this approach, feedback linearization is used to drive the actuated degrees of freedom of the robot towards a lower dimensional hybrid zero dynamics manifold. This manifold is specified as the zero levelset of a configuration-dependent output vector and represents the motion of the robot in its underactuated degrees of freedom. Many current approaches for guaranteeing safe HZD control [4, 41, 66, 68, 65] rely on the Poincaré stability of a periodic limit cycle. In some of these approaches, safety is contingent on the feasibility of a real-time Quadratic Program (QP) [41, 66, 67, 68, 65] that is dependent on the underactuated coordinates of the system. Guaranteeing feasibility of this QP thus requires a bound on these degrees of freedom. So far, such a bound has relied on local limit-cycle stability, which precludes recovery behaviors that would leave the neighborhood of the limit cycle. Recent work has been done to extend the range of safe HZD behaviours beyond a single limit cycle neighborhood [63, 101, 102, 5]. In [63, 101, 102], the controller is allowed to discretely switch between a family of periodic gaits. Safety is then ensured using a dwell time constraint that limits how frequently switching can occur. In [5], a combination of HZD and finite state abstraction is used to safely regulate forward speed of a fully-actuated bipedal robot.

1.3 Contributions

This dissertation can be broken into five contributions.

In the first contribution of this dissertation (Chapter II), I find the energetically optimal periodic motions for a high-fidelity model of RAM*one* across a wide range of forward speeds. The optimal motions are found using multiple-shooting trajectory optimization. At low speeds, the resulting optimal motions are found to share a striking resemblance to human and animal walking. At high speeds, the optimal motion abruptly switches to a gait similar to human and animal running. These resemblances indicate that the optimal motions of this robot match the inverted pendulum template of walking and the spring-loaded inverted pendulum template of running.

In the second contribution of this dissertation (Chapter III), I investigate the energetic performance of the real robot RAM*one* and demonstrate the trade-off between economy and stability. This investigation takes the form of a hardware study [89], in which a virtual-model controller is implemented on the robot in both hardware and simulation. The simulation performance is first compared to hardware results. The walking height for this controller is then varied and the cost-of-transport is evaluated. Straight-legged walking is found to be energetically economical for the robot RAM*one*, in agreement with expectations. In this study, energetic economy is found to continually improve as body height increases, up until the robot reaches a point of instability and falls. Avoiding falls is thus shown to take the role of a *constraint* when exploring within the space of gaits.

In the third contribution of this dissertation (Chapter IV), I construct a semi-autonomous safety regulating controller that guarantees the satisfaction of the "not falling" constraint. Such a regulator allows for free exploration of robot motions without risking the potential damage of a fall. The safety regulator is constructed by first finding a safe set of states for the robot, then regulating the input such that the robot state remains within this set. This method is applied to a 4 dimensional compass-gait walking model with uncertain dynamics, and we show that it can achieve a wide variety of walking motions while still remaining safe.

In the fourth contribution of this dissertation (Chapter V), I extend the safety regulator to more complex walking robots. This is challenging, as the sums-of-squares approach used for generating the safe sets in the previous contribution does not scale well with state-space dimension. I mitigate this scaling problem by constraining the safe set to lie on a low dimensional hybrid zero dynamics manifold. In order to add

flexibility to the robot behaviour on this manifold, a degree of actuation is introduced into the previously unactuated manifold. The safety regulator then takes controls this degree of actuation in order to preserve safety when needed. The method is validated on a 10-dimensional model of the bipedal robot Rabbit walking on flat terrain with continuously controllable torso pitch.

In the fifth contribution of this dissertation (Chapter VI), I propose an extension of the previous method towards more complex environments, using a reachability-based gait selector. This selector switches discretely between different gaits, using sums of squares analysis in order to maintain safety guarantees during switching. I then outline how such an approach can be applied to a rabbit model walking on rough terrain.

CHAPTER II

The Energetic Benefit of Robotic Gait Selection: A Case Study on the Robot RAM*one*

This chapter investigates the nature of economical locomotion for a realistic robot model in a simulated environment. A family of energy minimizing periodic motions is found for this model. The underlying properties of these motions are then analyzed, and parallels are drawn to fundamental principals of biological gait. The contents of this chapter were published in Robotics and Automation Letters 2017 Volume 2 [88], and are presented here as originally published.

2.1 Introduction

Being able to move in an energetically economical fashion over a wide range of velocities is a desirable property for legged robotic systems. Imagine, for example, an autonomous search and rescue robot. This robot would ideally operate in at least two distinct locomotor modes: a fast traveling mode to quickly get to the disaster scene, and a slow exploration mode that is employed at the scene while searching for survivors. Efficiency is key in either mode to maximize the operation time and range of the robot.

Similarly, biological systems need to move efficiently over a wide range of speeds. They achieve this energetic economy by using different gaits, such as walking, running, or galloping. By switching between gaits, animals and humans travel across a large range of speeds in an energetically economical manner [40, 57]. This chapter asks if the ability to change gaits can lead to a similarly improved energetic economy in a bipedal robot and investigates the nature of these gaits.

One way to gain insight into the mechanisms that govern locomotor economy is to study the characteristic properties of biological gaits. The simplest such property

is the contact pattern; that is, the sequence in which feet strike and leave the ground [39]. Another property consists of the phase relationship of kinetic and potential energy during locomotion. This phase relationship has been shown to be distinctly different at low and high speeds across a large range of species [16]. Another such property which characterizes gait is the shape of the contact force profiles, which has been used to classify gaits into walking or running [3]. In fact, as locomotion speed varies in animals, the type of locomotion (as classified by each of these gait characteristics) also changes [40, 16, 3].

If economical locomotion in robotic systems similarly varied as a function of speed, controller design would be profoundly impacted. Since different gaits constitute distinct motions which do not continuously transition from one to another, economical robot locomotion would require controllers that discretely switched from one gait to another. As a result, the question of the existence and the energetic benefits of distinct gaits is fundamentally important to the robotics community.

The extension of gait from biology to robotics is nontrivial, since biological and artificial systems are fundamentally dissimilar. To establish a meaningful connection between nature and robotics, a number of simple models have been proposed that provide a useful interface between biology and machine. They distill fundamental principles from locomotion in nature to provide templates for design and control of robots. Such models include a point mass on massless legs [94], passive dynamic walkers [59], the spring loaded inverted pendulum [12], and elastic walking models [29, 28].

To strengthen the link between these simple models and actual robotic systems, this chapter investigates a set of optimal motions for a specific robotic system (Fig. 1.1). RAM*one* is a bipedal robot, designed specifically to investigate energy efficiency and the role of gaits in robotic hardware. The robot is driven by high compliance series elastic actuators that can store large amounts of elastic energy and thus enable locomotion that exploits natural dynamics [77]. This chapter investigates properties of the energetically optimal motion at different speeds and determines whether using distinct gaits is useful for a robotic system and whether certain gait characteristics from biology or conceptual models are applicable to RAM*one*. Though this work focuses on illustrating the benefits of using distinct gaits at varying speeds on this particular robot, the presented approach clears the path for future hardware experiments since the developed model is realistic. In fact, robots with similar design and actuation will show similar characteristics and the proposed methods can be extended to other legged robotic systems.

The investigation in this chapter can be seen as an extension of previous work in optimal gait for conceptual models [93, 77, 107, 108] and builds upon optimization methodologies put forward in [17, 19, 109]. This chapter presents optimization across a variety of speeds for two distinct contact sequences and two different knee orientations. In addition to the added detail and realism that comes from modeling an actual robot rather than a contrived conceptual model, some of the important differences between this and previous work stem from the inclusion of articulating knees with nonlinear springs, and the presence of a large reflected inertia in the motors.

The remainder of this chapter introduces the model, the cost function, and the employed optimal control approach (Section 2.2), and classifies and discusses the obtained motions (Sections 2.3 and 2.4). In particular, we show that for RAM*one* the most efficient motions are ballistic walking at slow speeds and spring-mass running at high speeds. Furthermore, we show that it is clearly beneficial for RAM*one* to run with its knees pointing backwards. These results are put in context with findings from nature and with prior work on simple models. To the best of our knowledge, this is the most realistic model of a robotic system for which the benefits of changing gait have been demonstrated.

2.2 Constructing Optimal Motions

This chapter exploits trajectory optimization to discover energetically economic motions for the robot RAM*one* (Fig. 1.1). RAM*one* is a bipedal, five link, planar robot with circular feet and high-compliance series elastic actuators that are driven by brushless DC motors. It has the same legs as the Scarl*ETH* and Starl*ETH* robots [44, 78]. Since the robot has articulated knees, the presented approach considers motions for a pair of discrete morphologies: knees forward and knees backward with respect to the direction of motion. In addition, two different footfall patterns are evaluated. The first is a walking sequence with alternating single support and double support phases. The second is a running sequence in which phases of single support alternate with flight phases.

2.2.1 Model

The kinematic configuration, q, of the model of RAM*one* was described by the main body position x and y, the main body orientation ϕ, the hip angles α_L and α_R, and the knee angles β_L and β_R. Since the robot is driven by series elastic actuators, four additional coordinates encoded the motor positions $u_{\alpha L}$, $u_{\alpha R}$, $u_{\beta L}$, and $u_{\beta R}$. A

vector of motor torques T constitutes the input to this model. In the supplementary documents[1], MATLAB code that defines the dynamics, cost, and constraints of this model is included. The following is an overview of the model, highlighting only important features.

2.2.1.1 Dynamics

To formulate the dynamics, a hybrid dynamic approach is employed[30] in which continuous dynamics are interrupted by discrete events, corresponding to feet gaining or losing contact with the ground. The mechanical dynamics $\ddot{q} = f_c(q, \dot{q}, \tau)$, were derived using implicitly constrained Newton-Euler equations. Here τ represents the torques exerted on the joints by the actuators. In this approach, the generalized accelerations $\ddot{q}$ and the contact forces λ are simultaneously solved for.

If a foot leaves the ground, the contact is removed from the set of active constraints which changes the structure of the mechanical dynamics but does not alter the state. When a foot comes into contact with the ground, a new constraint is established, and the foot is assumed to be instantaneously brought to a halt by a collision impulse Λ. To this end, a discrete state transition $\dot{q}^+ = f_d(q^-, \dot{q}^-)$ is computed which expresses the generalized velocity after the event ($\dot{q}^+$) based on the pre-impact state (q^- and $\dot{q}^-$). When solving for $\dot{q}^+$ and Λ, one must carefully choose a post impact active constraint set that satisfies nonpenetration and nonnegative ground contact force conditions across the transition. In particular, this means that for the walking sequence (in which one foot is on the ground when the other collides), two outcomes are possible. Either the first foot remains on the ground, leading into an extended double-support phase, or the first foot immediately lifts off, creating an instantaneous double-support phase.

2.2.1.2 Series Elasticity

RAM*one* has two different types of series elastic springs, one type for the hip joints and the other for the knee joints [44]. The hip springs behave linearly with viscous damping and negligible dry friction. The resulting force model is:

$$\tau_\alpha = -k_\alpha \Delta\alpha - b_\alpha \Delta\dot{\alpha}, \tag{2.1}$$

where $\Delta\alpha = \alpha - u_\alpha$ is the difference between joint position and motor position, and τ_α is the torque on the hip joint.

[1] `https://bitbucket.org/ramlab/ral_2016`

To improve knee angle control while the foot is in the air, the robot's knees are designed with "endstops" in the series elastic knee springs [44]. The resulting knee joint is modeled as a position and velocity dependent spring damper system. When the knee joint is pushing against the endstop, we have a high stiffness and damping (k^1_β, b^1_β), otherwise they take on smaller values (k^2_β, b^2_β). The dry friction (f_β) observed in experiment is added to the model to obtain the following nonlinear spring:

$$\begin{aligned}\tau_\beta = &- k^1_\beta \Delta\beta - (k^2_\beta - k^1_\beta)\min(0, \Delta\beta - \beta_{sm}) \\ &- b^2_\beta \Delta\dot{\beta} - (b^1_\beta - b^2_\beta)\Delta\dot{\beta}\mathbb{1}_{(\Delta\beta-\beta_{sm}>0)}\mathbb{1}_{(\Delta\dot{\beta}>0)} \\ &- f_\beta \operatorname{sign}(\Delta\dot{\beta}),\end{aligned} \tag{2.2}$$

where $\mathbb{1}_A$ is the indicator function over the set A, $\Delta\beta = \beta - u_\beta$ is the difference between the joint and motor position, and τ_β is the torque on the knee joint.

2.2.1.3 Motor Model

Each joint of RAM*one* is actuated by a brushless DC motor that is connected via a gearbox and chain drive to a series elastic spring. In the model considered in this chapter, the motors are represented by their rotor inertia J_{rot}, the motor speed-torque gradient S_{mot}, and the net gear ratio of n_α and n_β for the hip and knee, respectively. Motor torques T and speeds $\dot{u}$ are limited by the maximum rated continuous motor torque T^{max} and the maximum rated input speed of the gearboxes $\dot{u}^{max}$. In the series elastic actuators, the motor accelerations $\ddot{u}$ are determined individually for each actuator:

$$n^2 J_{rot}\ddot{u} = (T - \tau). \tag{2.3}$$

All model parameters needed to compute the dynamics, cost, and constraints are provided in the supplementary MATLAB script `Parameters.m`. The parameters were identified and refined iteratively as the robot hardware was undergoing continuous testing and evaluation. The inertial properties were established from CAD models of RAM*one* and verified through measurement where possible. Other parameter values were determined using manufacturer specifications when available and were otherwise identified through direct or indirect measurement and fitting.

2.2.2 Optimization

Given this model, the following constrained optimization problem is solved to find the energetically economical periodic motions:

$$\min_{q,\dot{q},u,\dot{u},T,t_F} \text{CoT}\,(q,\dot{u},T,t_F) \tag{2.4a}$$

$$\text{s.t. Continuous Dynamics}\,(q,\dot{q},u,\dot{u},T) \tag{2.4b}$$

$$\text{Actuator Limits}\,(u,\dot{u},T) \tag{2.4c}$$

$$\text{Joint Limits}\,(q) \tag{2.4d}$$

$$\text{Foot Nonpenetration}\,(q) \tag{2.4e}$$

$$\text{Positive Contact Force}\,(q,\dot{q},u,\dot{u}) \tag{2.4f}$$

$$\text{Discrete Dynamics}\left(q^-,\dot{q}^-\right) \tag{2.4g}$$

$$\text{Foot Touchdown}\left(q^-,\dot{q}^-\right) \tag{2.4h}$$

$$\text{Positive Contact Impulse}\left(q^-,\dot{q}^-\right) \tag{2.4i}$$

$$\text{Foot Liftoff}\left(q^-,\dot{q}^-\right) \tag{2.4j}$$

$$\text{Periodicity}(q(0),\dot{q}(0),u(0),\dot{u}(0), q(t_F),\dot{q}(t_F),u(t_F),\dot{u}(t_F)) \tag{2.4k}$$

$$\text{Fixed Speed}\,(q,t_F)\,. \tag{2.4l}$$

The motion obtained from this formulation represents an energetic optimum for uninterrupted periodic locomotion without external disturbances, model errors, and sensor noise. In particular, the results will not take into account the additional energetic cost associated with stabilizing feedback.

2.2.2.1 Cost Function

The cost function used in the optimization estimates the electrical work required to drive the motors. Its computation reflects the fact that the motors share an input voltage rail, such that electrical power generated from one motor can be directly consumed by the other motors. The total power is thus the sum of the mechanical motor power and the motor copper losses of all four motors i. Since the robot has no means to store excess electrical energy in batteries or capacitors, if all motors together create negative net power, this power is dissipated in the form of shunt losses. Negative values were thus excluded when power was integrated over a full

stride with period t_F:

$$c = \int_0^{t_F} \max\left(\sum_{i=1}^{4} T_i \dot{u}_i + \frac{T_i^2}{n_i^2 S_{mot}}, 0\right) dt. \tag{2.5}$$

To compare energetic economy across different velocities, the resulting work was normalized to yield a dimensionless "cost-of-transport" (CoT) [27]:

$$CoT = \frac{c}{mg\Delta x}, \tag{2.6}$$

where Δx is the distance traveled in the stride and mg is the total weight of the robot.

2.2.2.2 Constraints

Mathematical constraints were used to ensure that the resulting motion was feasible on RAM*one*. These constraints fall into three categories: continuous constraints (Eqs.2.4b,2.4c,2.4d,2.4e,2.4f) which must be satisfied throughout the continuous phases of the trajectory, discrete constraints (Eqs.2.4g,2.4h,2.4i,2.4j) which must be satisfied during the event phases, and endpoint constraints (Eqs.2.4k,2.4l) which must be satisfied at the start and end of the trajectory. They encode the dynamics and physical limitations of the model, as well as the required periodicity and locomotion speed.

2.2.2.3 Optimizer

The constrained optimization problem in Eq. (2.4) was solved with the optimization package MUSCOD [13, 54, 22]. MUSCOD is a multistage multiple shooting optimizer that can perform simultaneous optimization of trajectories and control inputs for nonlinear systems with a predefined schedule of continuous modes. The trajectory and input are discretized as trajectory nodes with piecewise constant control inputs. Between nodes, the dynamics are integrated forwards with variable step integration, providing feasible trajectories even with coarse discretization. Additionally, MUSCOD allows for nonlinear state and input constraints throughout the trajectory.

Optimizations were performed for two distinct footfall sequences: a walking sequence and a running sequence. For the walking sequence, two possible collision outcomes had to be accounted for. In one outcome, the stance foot remains on the ground when the swing foot impacts which results in a double stance phase of finite

duration. In the other, the stance foot immediately lifts off, resulting in a zero duration double stance phase. Also, rather than implementing two models for the different knee directions, optimizations were conducted for positive and negative velocities.

Since this optimization is not necessarily convex, initialization is an important consideration when searching for global optima. The search for walking sequence trajectories was initialized with a feasible walking gait generated with a hand built controller. This gait contained a significant double support phase. The search for running sequence trajectories was initialized with a stationary trajectory. Once an optimal trajectory was found at a single velocity, this trajectory was used to initialize optimizations at higher and lower velocities. This process was repeated recursively. Optimizations were undertaken for the running and walking sequences and with the knees pointing forwards and backwards over a range of locomotion velocities with a velocity resolution of $0.02\,{}^{\mathrm{m}}/_{\mathrm{s}}$.

2.3 Optimal Motions and Gaits

Results of the optimizations are presented in Fig. 2.1. For a range of speeds, each curve shows the cost of transport (CoT) of the optimal motion for a given sequence and knee orientation. Note that attempts to enforce a non-zero duration double-stance in walking inevitably led to a higher CoT (Fig. 2.1). Hence, this chapter will focus solely on the walking sequence solutions with an instantaneous transfer of support.

At low speeds, motions with a walking sequence are found to be optimal, while at high speeds a running sequence consumes less energy. The transition happens at $1.04\,{}^{\mathrm{m}}/_{\mathrm{s}}$. Below this speed, three of the four motions exhibit a ballistic walking gait (Sec. 2.3.1), while above this speed the two running sequence motions exhibit a spring mass running gait (Sec. 2.3.2). The differences in cost are small below the transition speed, but clearly deviate for higher speeds. At these higher speeds, a significant benefit of having the knees pointing backwards rather than forwards is observed (Sec. 2.3.4). Motions with a walking sequence were identified for speeds as low as $0.12\,{}^{\mathrm{m}}/_{\mathrm{s}}$, while no running sequence for velocities lower than $0.50\,{}^{\mathrm{m}}/_{\mathrm{s}}$ with the knees pointing forward and lower than $0.90\,{}^{\mathrm{m}}/_{\mathrm{s}}$ with the knees pointing backwards were identified.

At a velocity of $1.04\,{}^{\mathrm{m}}/_{\mathrm{s}}$ a clear transition point in terms of the CoT-optimal footfall pattern is observed, changing from a walking sequence to a running sequence. For motions with a running sequence, the rate of increase of the CoT was drasti-

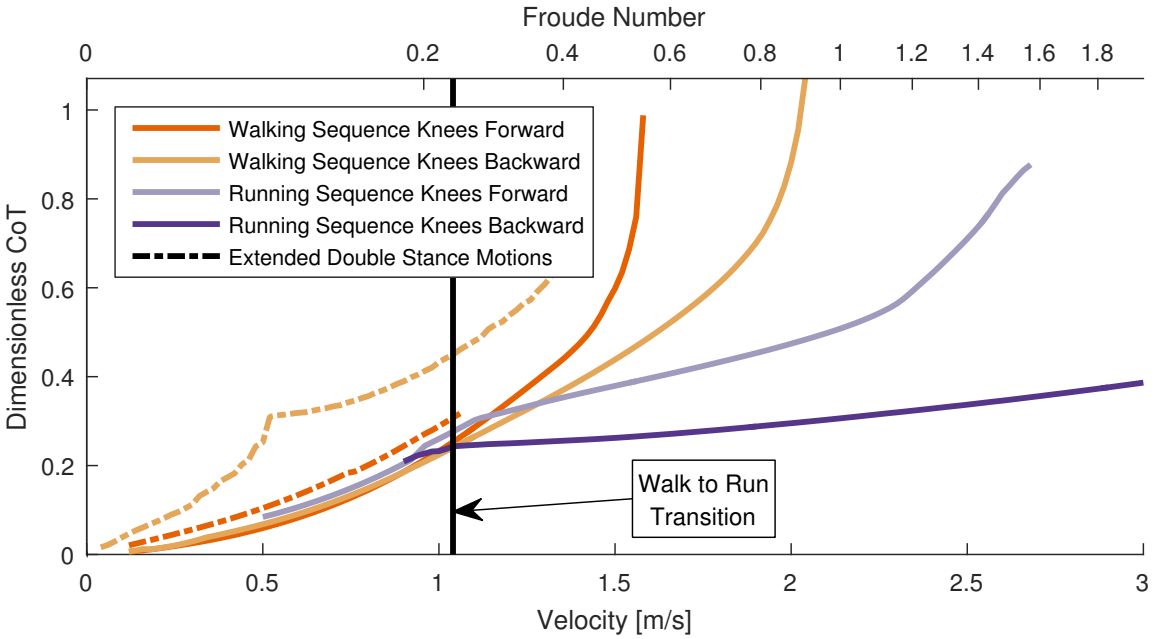

Figure 2.1: Cost of transport is shown as a function of speed given in $^{\mathrm{m}}/_{\mathrm{s}}$ and as a non-dimensional Froude number: $F = \frac{v^2}{g\ell}$ (where $\ell = 0.47\,\mathrm{m}$ is the extended leg length of the robot). At all speeds, the walking sequences with an extended double stance phase were suboptimal compared to those with instantaneous support transfer. As such, the remainder of the analysis focuses on walking sequences with zero double support phase. At low speeds, all four remaining cases had a similar CoT. Walking footfall sequences had a slight energetic advantage. For speeds below $0.88\,^{\mathrm{m}}/_{\mathrm{s}}$, the walking sequences with the knees pointing forward had the lowest CoT, for speeds between $0.88\,^{\mathrm{m}}/_{\mathrm{s}}$ and $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$ it became optimal for knees to point backwards. Motions with a running footfall sequence were slightly more energetically expensive below $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$. However, their CoT was only up to 0.036 higher. At high speeds above $1.04\,^{\mathrm{m}}/_{\mathrm{s}}$, the CoT of the four gaits diverged. Moving with a running sequence and with the knees pointing backwards became by far the most efficient mode of locomotion. Using a walking sequence increased the CoT by up to 0.77 (259 %) at $2.04\,^{\mathrm{m}}/_{\mathrm{s}}$. Having the knees point forward instead of backward increased the CoT of the running sequence by up to 0.52 (148 %) at $2.68\,^{\mathrm{m}}/_{\mathrm{s}}$.

cally reduced beyond this transition point. In contrast, the CoT of motions with a walking sequence kept growing at an increasing rate. As detailed in the subsequent sections, this divergence corresponds to a sudden and discrete transition in terms of the underlying gait characteristics.

2.3.1 Ballistic Walking at Speeds Below 1.04 $^{m}/_{s}$

Motions with a walking sequence were energetically the most efficient at speeds below 1.04 $^{m}/_{s}$. When examining the exchange of kinetic, gravitational, and elastic energy content over the course of a stride, the motions exhibited a clear *out of phase* transfer between kinetic and gravitational energy (Fig. 2.3). Additionally, gravitational energy was highest during mid-stance with the center of mass moving in an upwards arc. There was virtually no elastic energy storage in motions with the knees pointing forward. Some energy was stored elastically in motions with the knees pointing backwards. The ground reaction forces (Fig. 2.4) were mostly flat apart from a large peak at the beginning of single stance (an effect of the contact collision). With the knees pointing forward, a peak in force towards the end of stance, reminiscent of a human push-off, was discovered. This is consistent with the demonstrated energetic benefit of pre-emptive push-off for both animals [35] and machines [82].

The characteristics of the energy exchange are clearly indicative of *ballistic walking* [61]. Especially with the knees pointing forward, the robot's optimal motion is much like that of a compass gait walker [46] with almost no energy storage in the elastic actuators and the dynamics represent those of an inverted pendulum. This is in stark contrast to earlier studies with simplified models of robotic systems, which found that at low speeds, elastic walking with substantial elastic energy storage in leg springs and an extended double support was the optimal locomotion mode [108]. The absence of elastic walking gaits [29, 108] in RAM*one*'s motion is likely a consequence of the fact that the knees are nearly straight throughout stance (Fig. 2.2) which makes the robot's legs rigid. This knee extension is not as pronounced when the knees are pointing backwards and more storage of elastic energy was observed in this configuration. While the two walking sequences shared many conceptual properties, there remained some visible differences including varying amounts of elastic energy storage throughout the gait, straightness of the knees, stride length, and stride frequency. However, overall both motions were clearly ballistic walking gaits.

For motions with a running sequence, the outcome was more structurally dependent on the knee direction. With the knees pointing backwards, the duration of the air-phase was nearly zero and the energy dynamics showed an exchange between ki-

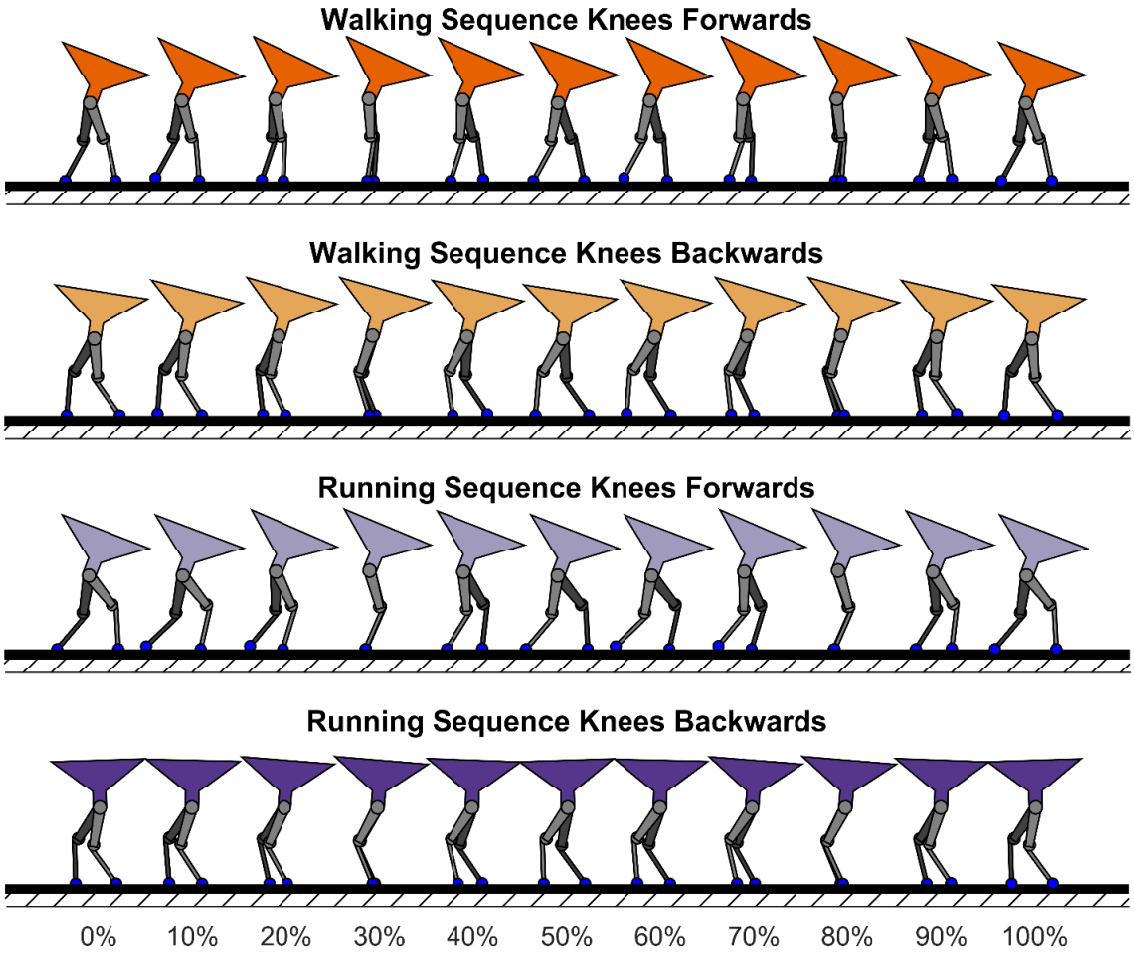

Figure 2.2: Trajectory stills show the optimal motions at a speed of $0.9\,{}^{\mathrm{m}}/_{\mathrm{s}}$. This speed is below the walk to run transition and the gaits share similar properties that are indicative of ballistic walking. In particular, the knees are extended, the main body is pitched forward, and the center of mass moves in an upwards arc. The running sequence with knees pointing backwards is an exception to this pattern and shows characteristics more reminiscent of Groucho running.

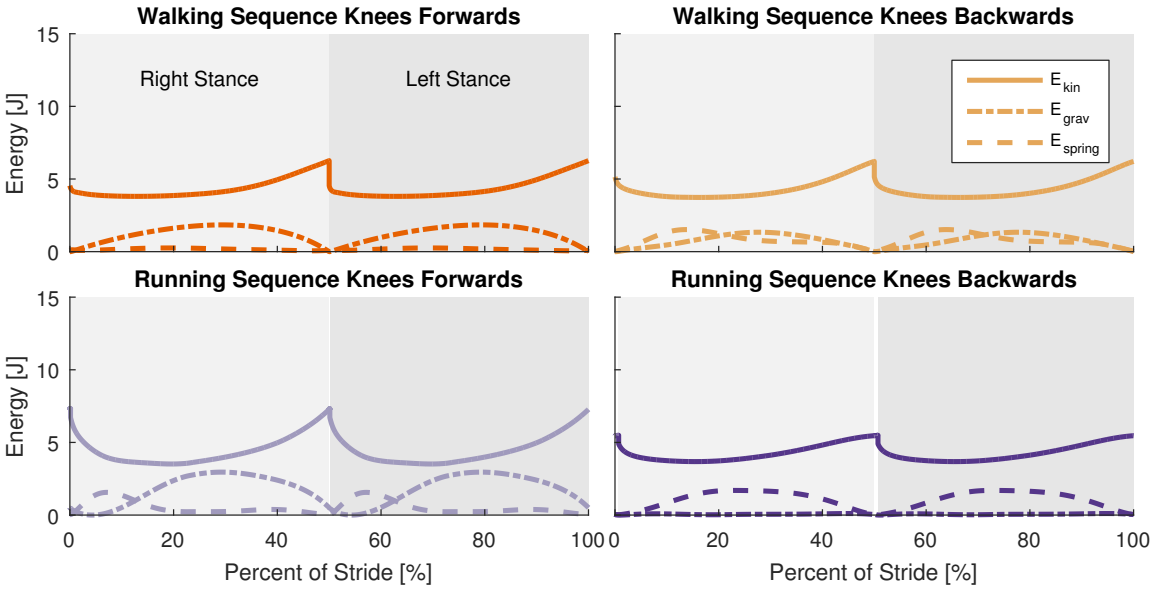

Figure 2.3: Center of mass kinetic, gravitational, and elastic energy flow for all motions at 0.9 m/s. Both walking sequence motions and the knees forward running sequence motion exhibit an out-of-phase relationship between kinetic and potential energy. This is prototypical of ballistic walking [61]. The knees backwards running sequence gait exhibits Groucho running [60], characterized by a constant potential energy and an out of phase exchange between kinetic and elastic energy.

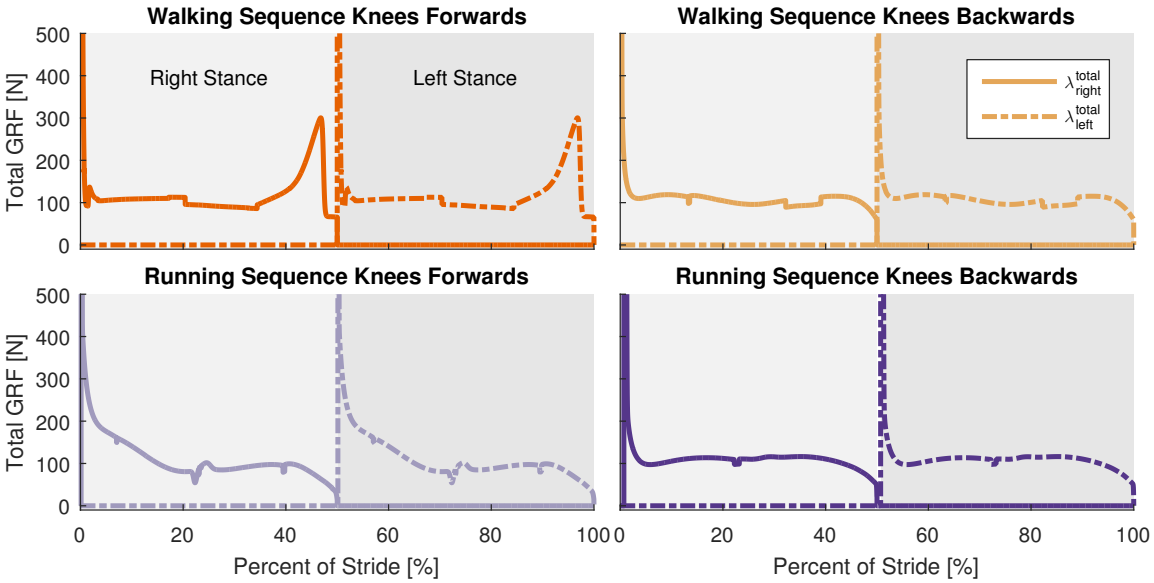

Figure 2.4: Ground reaction force magnitudes at 0.9 m/s. A distinct pushoff force is observed near the end of each single stance phase for the knees forward walking sequence. This pushoff force is conspicuously absent in the other three motions.

netic and elastic energy storage, with hardly any fluctuations in gravitational energy (Fig. 2.3). This gait could be described as *Groucho running* [60]. However, only such motions for speeds larger than 0.90 m/s could be found. In contrast, with the knees pointing forwards, such motions were found for speeds as low as 0.50 m/s. These motions showed all characteristics of ballistic walking, even though they had a running footfall sequence. Characteristics included an out-of-phase relationship in kinetic and gravitational energy (Fig. 2.3), an upwards arc of the center of mass, and extended knees (Fig. 2.2). The optimizer brought the air-phase duration of these gaits nearly to zero (Fig. 2.5), getting as close as possible to an instantaneous transfer of support.

One should note that there is an important difference between a walking sequence with a zero duration double support and a running sequence with a zero duration air-phase. In the walking sequence, the touchdown impact at the leading leg and the resulting impulse that propagates through the mechanical structure create an immediate lift-off of the trailing leg. When we enforce an air-phase (even of vanishing time duration), the trailing leg must leave the ground before the impact collision, meaning that lift-off must be generated without the assistance of the ground impulse. Despite this important difference, the optimizer still converged to a ballistic walking motion.

2.3.2 Spring-Mass Running at Speeds Above 1.04 m/s

As locomotion speed was increased beyond 1.04 m/s, a sudden and significant increase in the air-phase duration was observed for motions with a running footfall sequence (Fig. 2.5). With knees pointing backwards, for example, the air-phase duration (as percentage of a full stride) increased in a single velocity increment from 1.3 % at 1.04 m/s to 18.6 % at 1.06 m/s. The sudden increase is indicative of a structural change in the motion strategy. For speeds larger than 1.04 m/s, the energy flows of these motions exhibited an *in phase* relationship between kinetic and gravitational energy, with energy being stored as elastic energy in the actuator springs (Fig. 2.7). Gravitational energy was lowest at mid-stance with the center of mass moving in a downwards arc. For both knee orientations, the ground reaction forces showed an initial (collision) peak followed by a single hump (Fig. 2.8).

These characteristics are clearly indicative of *spring-mass running*, also called *spring loaded inverted pendulum (SLIP) running* [12]. The legs are used as springs and the motion resembles that of a bouncing ball. The required leg-compliance is achieved by a more pronounced knee bend that softens the leg (Fig. 2.6).

In contrast to the sudden changes that were observed in motions with a running

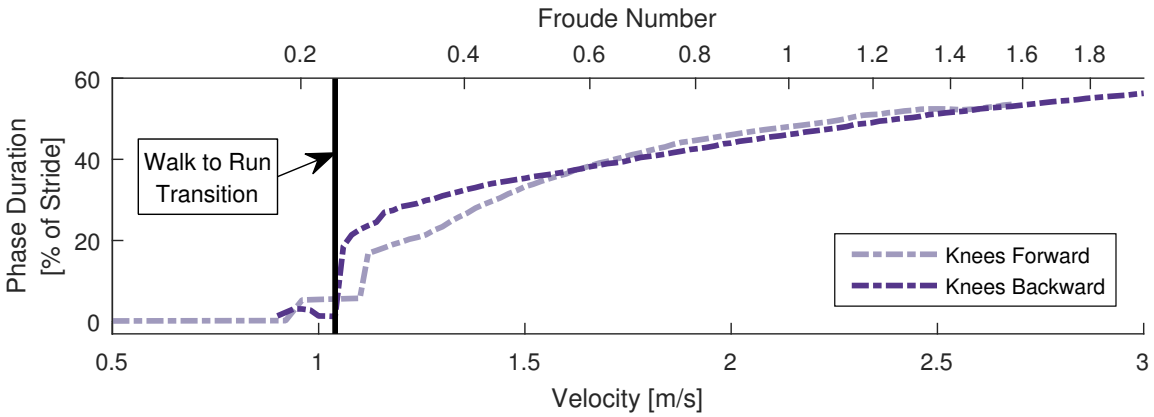

Figure 2.5: Flight phase durations of the running sequences. The optimal flight phase duration is near zero for speeds below the walk to run transition and jumps up sharply at speeds above this point, indicating a sudden and discrete change in gait.

sequence, motions with a walking sequence remained mostly unchanged across the transition point. In particular, they continued to exhibit an instantaneous transfer of support, extended knees, and all other characteristics of a ballistic walking gait.

2.3.3 Walk to Run Transition

At the transition point, the most economical footfall sequence changed from a walking sequence to a running sequence. This change in footfall sequence was not the only indication of gait change. When considering only motions with a running sequence, we observed a clear structural change at this speed. Below this speed, the motion with forward knees closely resembled a ballistic walking gait, despite the presence of an enforced air-phase. The energetic benefits of changing gait thus likely do not originate primarily in the footfall sequence, but more in the dynamic pattern of the chosen gait.

The transition occurred at a Froude number of 0.23. This is similar to the 0.25 observed for a simple biped in [108], but is significantly smaller than the 0.42 observed for humans [94]. Additionally, the transition speed is similar for both forwards and backwards knee directions. The similarity of the transition speed between the two knee orientations and the prismatic legs of [108] suggests the existence of an underlying trigger that is independent of leg morphology.

What is this mechanism that drives the sudden structural change in motion at speeds of 1.04 $^{m}/_{s}$? Why is ballistic walking more efficient at lower speeds and why is spring-mass running more efficient at higher speeds? One hypothesis is proposed in [94], where the authors found a similar transition for a minimal biped model. They

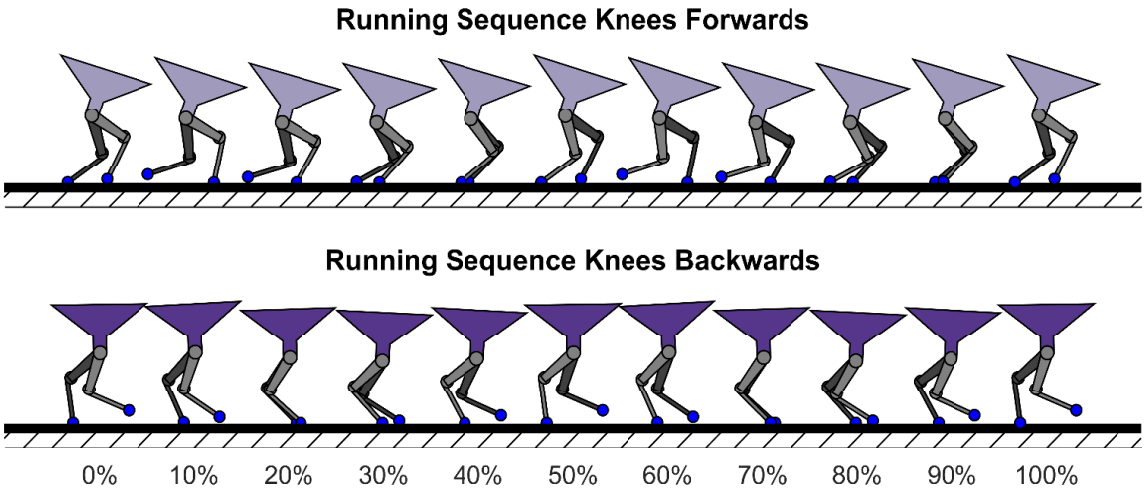

Figure 2.6: Trajectory stills of the optimal running sequence motions at 1.2 $^{m}/_{s}$. In contrast to the same footfall sequence at lower speeds (Fig. 2.2), the motions now exhibit clear properties of spring-mass running. This includes a pronounced knee bend and a downwards arc of the center of mass.

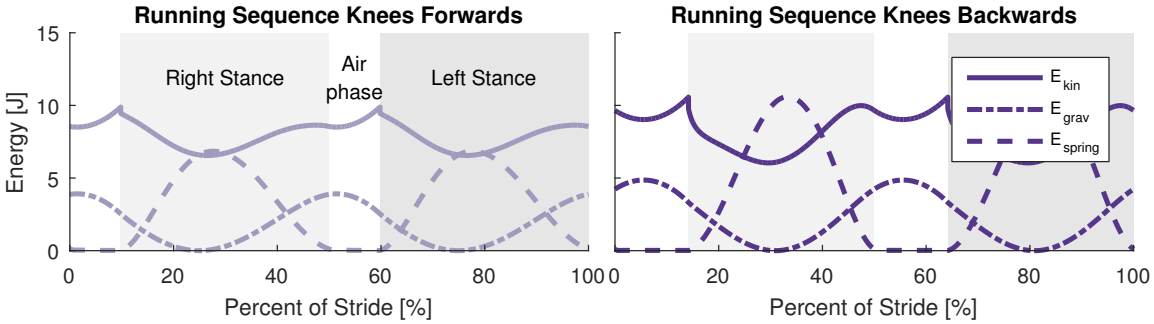

Figure 2.7: Center of mass kinetic, gravitational, and elastic energy flow for running sequence motions at 1.2 $^{m}/_{s}$. The left and right contact phases of the motion are indicated with a shaded background. Both motions exhibit the in-phase kinetic and gravitational energy oscillation with out of phase elastic energy that characterizes spring mass running [12]

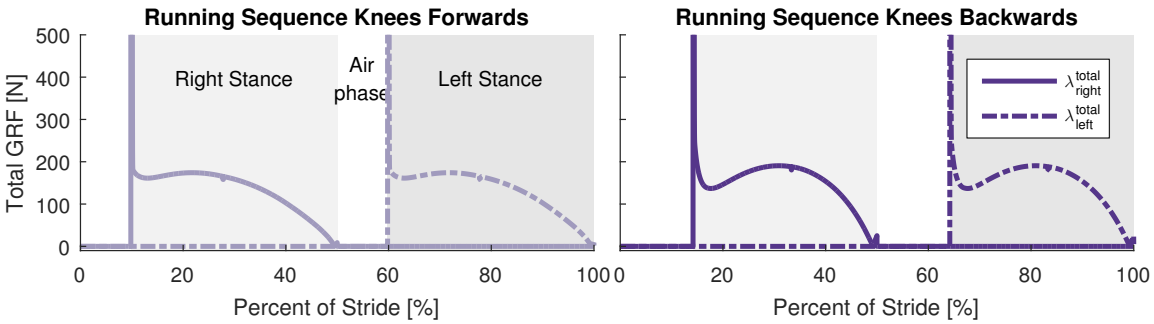

Figure 2.8: Total ground reaction force at each foot for running sequence motions at 1.2 m/s. Both knees forward and backwards motions exhibit a sharp initial force at contact followed by a single hump.

argued that this was driven by the large collision cost of high speed ballistic walking.

In order to see if similar arguments apply to our results, we separated the total CoT into losses due to collisions, damping losses, copper losses, and shunt losses. Since the motion trajectories were periodic, no energy was gained or lost by the robot over one period, and the energy required to drive the actuators matched the above mentioned losses:

$$c = Q_{coll} + Q_{damp} + Q_{copper} + Q_{shunt}. \tag{2.7}$$

To match the definition of the CoT, the values of these losses were scaled by $(mg\Delta x)^{-1}$.

This breakdown of contributors to overall cost was computed for motions with the knees pointing backwards in both a walking and a running sequence (Fig. 2.9). For the walking sequence, copper losses were the primary contributor to the cost (on average 57 %), followed by damping losses (31 %) and collision losses (13 %). A similar breakdown was obtained for the running sequence below 1.04 m/s. For the running sequence above 1.04 m/s, this ratio was inverted, damping losses led (on average 62 %), followed by copper losses (31 %) and collision losses (6 %). Negative electrical work appeared solely in high speed walking sequence gaits.

The fact that collision losses were dominated by damping and electrical losses in our cost function would indicate that the mechanism proposed in [94] is not the sole factor driving the walk to run transition.

2.3.4 Influence of the Knee Direction

A notable finding was the large difference in CoT as a function of knee direction. For the running sequence, moving with forwards knees required on average 60 % more energy than with backwards knees. This is in agreement with prior findings in [34],

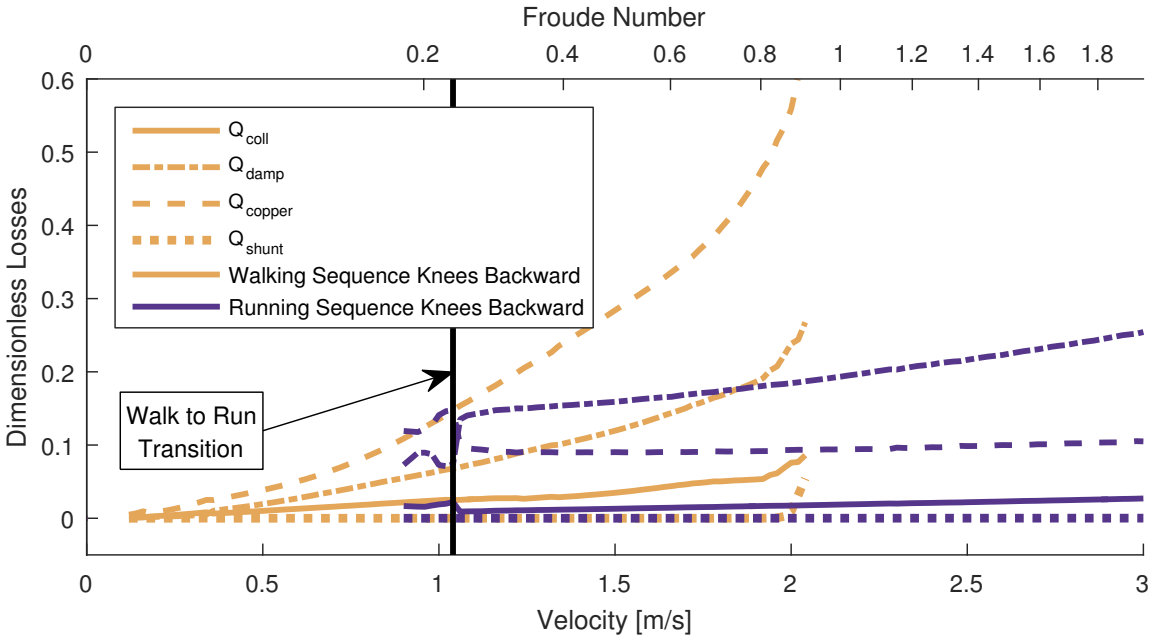

Figure 2.9: Comparison of loss contributions (normalized by $mg\Delta x$) between walking and running sequences . Results are shown across a range of speeds and for motions with the knees pointing backwards. All components of the walking sequence motion increase smoothly across speed, with electrical losses dominating. In contrast, the individual costs of the running sequence motion change discretely across the walk to run transition. Below the transition, the costs of the running sequence motion roughly follow those of the walking sequence motion. Above the transition, the electrical losses sharply decrease and damping losses dominate the cost. This sharp jump is a result of the transition from *Groucho running* to spring mass running.

in which it was shown that backwards kneed gaits are more efficient for most five link bipeds.

The increased cost for forward kneed running is incurred during leg retraction at the beginning of swing. Since the reflected rotor inertia for this robot is large in comparison to the leg mass, the majority of the cost of leg retraction comes from reversing the direction of the motors.

For the knee backwards gait, the push-off velocity is largely generated from the extension of the knee joint (kinematically governed by the liftoff speed). This velocity can be driven by the knee springs, allowing the motors to begin moving in flexion even before liftoff (Fig. 2.10). As a result, the knee rotors don't need to reverse direction during swing.

For the knee forwards gait, the push-off velocity is mainly generated from the extension of the hip. Ideally this velocity would come from spring deflection (as in the knee), however, this deflection would create a large hip torque before liftoff. Since torques in the hip before liftoff lead to pitching of the main body, the hip spring deflection must be small. This forces the hip motor to match the hip joint velocity (Fig. 2.10).

Therefore, knee forwards running gaits must quickly dissipate a large angular momentum stored in the hip motors at liftoff in order to retract the swing leg. In this case 1.13 J (0.0157 when expressed as a dimensionless energy) of hip rotor energy must be dissipated. When the knees are pointing backwards, only 0.02 J (0.0004) of energy is dissipated.

2.4 Discussion & Conclusion

This chapter presents a study on the question of economical gait selection for legged robotic systems. Optimal control was used to generate energy optimal motions for a realistic model of the bipedal robot RAM*one*. Two different footfall sequences (a walking sequence with a double-support phase and a running sequence with an air-phase) and two different orientations of the knee joints (pointing forwards and backwards) were compared. Actuator inputs and motion trajectories that minimized the electrical cost of transport were identified for a range of velocities. The optimal motion at each individual speed was computed and each such motion was characterized using established gait classifications.

At a speed of 1.04 $^{\mathrm{m}}/_{\mathrm{s}}$ the optimal gait was found to change from ballistic walking with an instantaneous double-support to spring-mass running with an extended

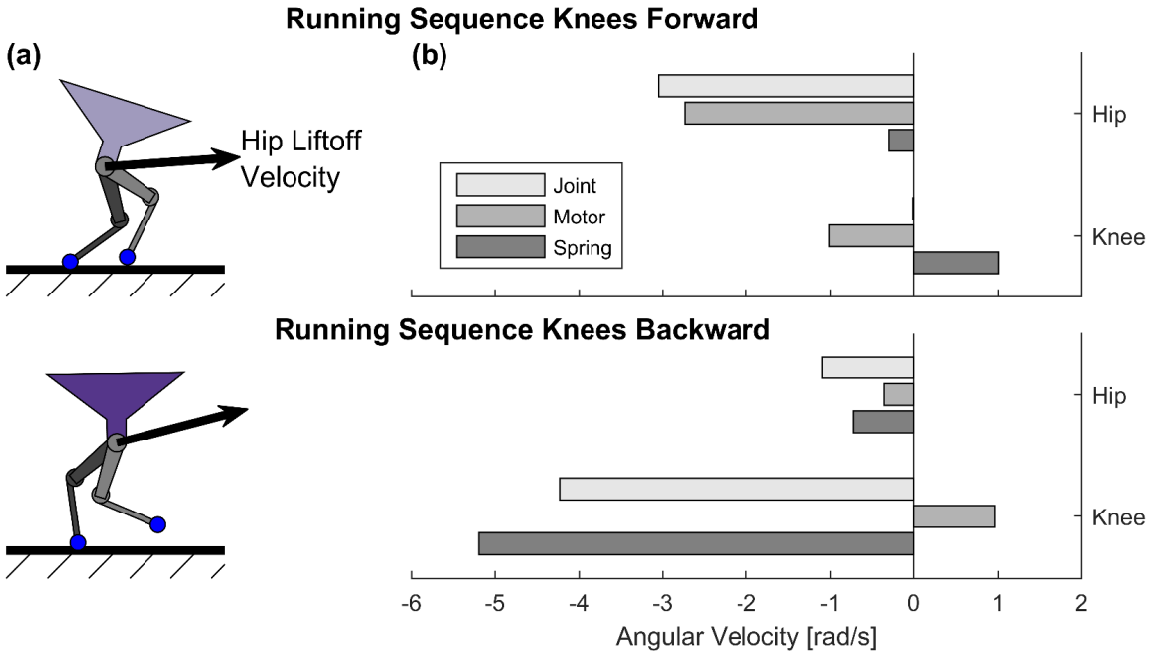

Figure 2.10: The different knee directions in the running sequence motion lead to different robot configurations at liftoff (shown in (a) for a velocity of 1.2 $^{\mathrm{m}}/_{\mathrm{s}}$). As a consequence, the joint velocities in the swing leg (shown in (b)) differ greatly between the two cases. These joint velocities are the sum of the motor and spring velocities. A significant deflection in the knee springs decouples the joint from the motor. Joint and motor velocities are thus distinctly different, and the knee motor is already moving to retract the leg while the knee joint extends for liftoff. The same does not hold for the hip joint, where joint and motor motion must be more similar and the motor motion can only be reversed after lift-off. Because of this, running with the knees backwards is at an advantage, since it requires a much smaller hip velocity. [Note that all velocities are defined to be positive when counter clockwise. A positive knee velocity thus indicates knee extension when the knees are forward, and knee flexion when the knees are backwards.]

air-phase. Switching from ballistic walking to spring-mass running reduced energy consumption by up to 88 %. *This result illustrates clearly that it is beneficial for RAMone to employ different gaits at different speeds.*

Notably the different motions at distinct speeds distinguished themselves primarily in terms of the dynamics of the motion. Gaits were clearly identified as either ballistic walking or spring-mass running. That is, at low speeds nearly no energy was stored in the actuator springs, while at high speeds almost all of the energy fluctuations within the robot were conducted through the springs. There was no continuous transition between these types, with a sudden change occurring at a speed of 1.04 $^{\mathrm{m}}/_{\mathrm{s}}$. This switch was not only initiated by a change of footfall sequence, as motions with a running footfall sequence showed clearly different dynamic behaviors below and above this speed.

That the identified optimal motions shared many properties with gaits found in humans and animals was not expected per-se. While the overall morphology of RAM*one* roughly resembles that of a human (when the knees point forward) or of birds (when the knees point backwards), there are considerable differences between this robot and biological systems. Among these, RAM*one* lacks ankles and feet, is actuated by electrical DC motors which have a considerable reflected inertia, has springs that are only slightly damped, and employs a cost function that trades-off work and force penalties specific to electric actuators. Still, the general trend of transitioning from ballistic walking to spring-mass running, as well as the main characteristics of the individual gaits (especially with respect to the exchange of potential, gravitational, and elastic energy) were similar. Most major differences, such as the lack of the double stance phase found in human walking, were likely a consequence of the rigidity of the robot's structure (which differs from the compliant legs of humans and animals).

It is tempting to interpret the backwards-knee orientation as a transformation of the knees into ankles, thereby making the morphology of the robot more similar to that of, for example, birds or ungulates. However, this interpretation is not made in this chapter since, unlike animals, the inertia of RAM*one*'s leg is largely dominated by the reflected inertia of the actuators (by a factor of 10). Still, the demonstrated benefits left no doubt that RAM*one* should locomote with knees pointing backwards.

It is necessary to note that the optimization problems in this chapter are not convex. Relying on local methods, we cannot guarantee that our results represent globally optimal motions. However, despite seeding each contact sequence and knee orientation with dissimilar initial trajectories, all four optimizations converged to similar gaits with similar cost of transport at low speeds. For the running sequence, this

means that optimizations seeded with locally optimal ballistic running gaits converged instead to pendular walking gaits at speeds below the walk to run transition. Since walking and running have discretely different trajectories, there is likely no sequence of locally optimal gaits that lead gradually from the first local minimum to the other. This indicates that our search is capable of escaping suboptimal local minima.

This work prompts several questions that merit further investigation. One is that of generality. How do these results extend to other robots? Another is gait transition. What drives the transition in RAM*one*, and why does it occur at similar speeds across leg morphologies? We also hope to understand specific features of the gaits, such as the significantly bent knee at liftoff and touchdown during knees forward running.

Additionally, while we did our best to model RAM*one* as precisely as possible, certain effects can never fully be encoded in a simulation. This includes gearbox friction, foot deformation, elastic forces in cables, and parametric model error. To truly appreciate the role and benefits of using different gaits in legged robots, one thus has to implement optimal gaits on an actual robotic prototype.

This study is another milestone in the effort to understand the meaning and the usefulness of different gaits in legged robotic systems. It extends on an understanding of biology [40, 16], simple passive models [29, 28] and conceptual models [108], and adds a strong layer of realism to this question. It now remains to explore how these results translate to actual physical robots.

CHAPTER III

RAM*one*: A Planar Biped for Studying the Energetics of Gait

This chapter moves our investigation from the ideal simulated world of the previous chapter towards physical reality. Here the real robot RAM*one* is used to conduct experimental evaluations of a simulation-inspired walking controller, as well as to validate our detailed robot model. The contents of this chapter were published at IROS 2017 [89], and are presented here as originally published.

3.1 Introduction

For legged robots to be practically useful, they have to be robust (able to operate in the presence of noise, model errors, external disturbances, and uncertain environments) and versatile (able to perform different tasks) while using little energy to function. However, no robot to-date demonstrates all of these qualities. Robots that seem to be most robust and versatile, such as Atlas [23] and Asimo [84], use powerful actuation for high-precision control. This enables them to perform fast and strong actions to react to and cancel large perturbations. But such control also makes the robot motions energetically costly. For example, Atlas has a cost of transport[1] (CoT) of ~ 5, which is 15 times that of a walking human [10]. In contrast, the robot Cornell Ranger is about as energy-economical as humans, with a CoT of 0.28 [10]. Ranger was designed and controlled specifically to walk with little power, *e.g.* it exploits natural dynamic motions and performs work when it is most optimal to do so ('preemptive push-off'). But Ranger is barely stable, as it falls on slopes of just two degrees and is only able to walk, but not, say, run or hop.

[1] Cost of transport is a standard metric of energetic economy [98], defined as the energy consumed per distance traveled normalized by body weight.

So how can we make robots that combine robustness, versatility and energy economy, which many legged animals seem to do naturally [98]? One approach is to use common attributes of versatile robots, such as sufficient and fast actuation in all joints, along with general principles that are considered beneficial for energy economy. Such principles influence three different levels of robot control and design:

- *Hardware components* that minimize energy losses in the system, regardless of the specific motion or controller. These include efficient actuators (*e.g.* DC motors instead of hydraulics), small friction between different elements, and regenerative power. The MIT Cheetah is one robot that effectively makes use of such hardware components [86].

- *Hardware design principles* that can be exploited by the controller for economical gaits. For example, light legs with small feet allow for swinging the legs quickly and with little effort [86]. Springs reduce ground-impact energy losses; they can also store energy generated by negative work and release it later to help power the robot motion [2]. The ATRIAS robot is an example of a robot that follows these principles [42].

- *Control principles* that optimize energy use during locomotion. For example, controllers that exploit underlying natural dynamics, such as pendular walking, can greatly reduce energy consumption in robots [92]. Additionally, the use of different gaits can allow for energy economical motion across a large range of speeds [40][94][88].

Our goal is to study energy-economical locomotion that is also versatile and possibly robust. For this purpose, we designed the planar bipedal robot RAM*one* (Fig. 1.1). RAM*one* is a bipedal version of the robot Scarl*ETH*, a monoped which was designed at ETH Zurich according to many of the above principles [44]. RAM*one* is similar to other planar bipedal robot platforms including ERNIE [109], KURMET [47], and MABEL [92].

In our past work [88], we determined the energy optimal motion strategies for a detailed model of RAM*one*. We found that for this robot, straight-legged pendular walking is the most economical gait at speeds below $1.04\,\mathrm{m/s}$. Above this speed, spring mass running was found to be optimal. These optimal motions are consistent with results for a simple biped model [94] and with the way humans prefer to move at different speeds [57]. However, simulation, no matter how detailed, is only an approximation of the real world, and such results do not always translate to actual

Hip Offset, l_H	0.138 m
Thigh Length, l_{L2}	0.200 m
Shank Length, l_{L3}	0.264 m
Foot Radius, r_{foot}	0.028 m
Main Body Mass, m_1	7.90 kg
Thigh Mass, m_2	0.79 kg
Shank Mass, m_3	0.57 kg
Left Hip Spring Constant, k_α	74 $^{\mathrm{Nm}}/_{\mathrm{rad}}$
Right Hip Spring Constant, k_α	74 $^{\mathrm{Nm}}/_{\mathrm{rad}}$
Left Knee Spring Constant, k_β	30 $^{\mathrm{Nm}}/_{\mathrm{rad}}$
Right Knee Spring Constant, k_β	32 $^{\mathrm{Nm}}/_{\mathrm{rad}}$

Table 3.1: RAM*one* Kinematic, Mass and Spring Properties

robots. For this reason, we have constructed a hardware platform to directly explore the question of economical motion in the real world.

In this chapter, we introduce the hardware of RAM*one*, we describe a virtual model controller which demonstrates stable walking across a range of body heights, and we present a realistic simulation of the robot which is shown to agree with the hardware results. In addition, we conduct an energy-economy study across body height in both simulation and hardware, which shows that straighter-legged walking is more economical for this robot.

3.2 Methods

3.2.1 RAM*one* Hardware

RAM*one* consists of a main body, two identical legs and a motion planarizer. A schematic of the robot is shown in Fig. 1.1.

The main body is designed to be lightweight while maximizing its moment of inertia. It mounts onto a carbon fiber tube that confines the motion to the sagittal plane and serves as its pitch axis, which is offset from the legs' hip axis. The main body center of mass is located at the pitch axis.

Each leg is based on the Scarl*ETH* design [44], with two joints, one each for flexion/extension of the hip and knee. Each joint uses linear compression springs in series with a Maxon EC60 motor mounted to a harmonic drive gearbox with a gear ratio of 50:1. The hip has two pre-compressed springs in an antagonistic setup yielding linear behaviour throughout the full range of motion. The knee places a unilateral damper in series with a pre-compressed spring yielding a non-linear behaviour. Force

control in the knee can only be achieved in one direction where compression of the spring behaves linearly. The unilateral damper restricts spring deflection in the other direction and passively attenuates undesired deflections. The motors and gearboxes are mounted together in one component that comprises the hip axis which keeps the overall center of mass near the hip and inertia of each leg segment to a minimum. Chain and cable pulley systems are used to actuate the joints' motion.

The motion planarizer is detailed in [33]. Separate horizontal and vertical sliders carry the carbon fiber tube used for mounting the robot. A motor-driven cable pulley system is able to catch the robot upon falling or provide a specified supporting force.

The controller for RAM*one* (see Section 3.2.2) is model-based. Thus it is important to have accurate estimates of RAM*one*'s kinematic, mass, and spring properties (Table 3.1). The kinematics of the robot were obtained through a mixture of CAD values and direct measurement. Each link was weighed to identify its mass and balancing measurements were used to estimate centers of gravity. The rotational inertias were obtained from the natural frequency when swinging freely. Hip and knee spring constants and knee spring pre-compression were measured separately for each leg by ranging spring deflection and measuring resulting contact force with a force transducer.

The motors are controlled with EPOS3 positioning controllers mounted on the main body that use an optical encoder and a hall effect sensor integrated in the motor to track the motor's position and speed. The motor torque is also reported by the motor controllers. High resolution single-turn absolute encoders are used to measure hip position and pitch and incremental encoders are used to measure knee deflection. Incremental linear encoders mounted to the planarizer are used to measure the robot's main body position (x and y).

Ground contact is sensed through a specially designed foot using limit switch contacts. An air filled racquet ball is affixed to the foot and serves as the external contact, providing cushioning and a high friction coefficient.

The motor controllers, sensors, motion planarizer and a treadmill are integrated together in a Beckhoff TwinCAT 3 PC-based control system that uses the EtherCAT network protocol. The motor controllers have built-in EtherCAT interfaces while the other sensors are integrated through EtherCAT modules connected to the network through an EtherCAT coupler. The EtherCAT modules and coupler, control computer and main power are located off-board and connected through a tether. Programming for RAM*one* is done with MATLAB/Simulink using MathWorks' xPC Target to create the real-time testing environment.

3.2.2 Controller

In order to make RAM*one* walk steadily while maintaining a given main-body height and forward speed, we designed a walking controller that is conceptually simple while allowing for flexible parametrization. The controller resembles Pratt's Turkey Walk controller [74], with an added phase parametrization to govern stance transitions. This high level controller sends its output to a set of low-level joint controllers, which, in turn, send velocity commands to the EPOS3 motor controllers.

3.2.2.1 Low Level Motor Controllers

In order to safely control body position over uneven terrain as well as precisely control foot position in the air, we use both position and force controllers at the joint level. These low level controllers are similar to those described in [44].

For torque control, we regulate the extension of the series elastic elements in the joints to achieve a desired force. This is done using a PD controller on spring deflection, with feed-forward terms consisting of the joint angle velocities. Due to the unilateral damping in the knees, our force control is only accurate above a knee torque of 1.5 Nm. To account for this, we saturate the knee torques at this threshold.

We use similar PD controllers with feed-forward terms to govern our low level joint position controllers. In the knee, the spring pre-compression keeps the joint pressed against the motor for small torques which allows for the use of a motor position controller to achieve a desired joint position. In the hip, we use a proportional state feedback controller on the joint and motor position and joint velocity to achieve a desired joint position.

3.2.2.2 Virtual Model Controller

The basis of our walking controller is a stance-dependent control target for the main-body center of mass position. This control target is achieved using a set of virtual forces acting on the main body: horizontal and vertical virtual forces F_x^{virt} and F_y^{virt}, and the virtual torque F_ϕ^{virt} about the main-body center of mass.

We use virtual forces to control body height y and pitch angle ϕ during both double stance and single stance:

$$\begin{aligned} F_y^{virt} &= K_y^P\left(\hat{y}-y\right) - K_y^D\left(\dot{y}\right) + m_1 g \\ F_\phi^{virt} &= K_\phi^P\left(\hat{\phi}-\phi\right) - K_\phi^D\left(\dot{\phi}\right). \end{aligned} \tag{3.1}$$

Here $\hat{y}$ and $\hat{\phi}$ are the desired height and pitch respectively. Note that we add the weight of the main body ($m_1 g$) to the vertical virtual force in order to offset gravitational forces.

During double stance, in addition to the pitch and height control, we aim for zero force in the horizontal direction:

$$F_x^{virt} = 0. \tag{3.2}$$

This approach differs from Pratt et al., who use a virtual damper [74] to control the forward velocity of the robot during double stance. We instead control the forward speed of the robot by adjusting the relative time spent by the main body behind or in front of the stance foot during single stance. This indirectly controls the energy added to the system during single stance. To adjust this, we heuristically tune the parameter Δ_x in the phase parametrization (see Sec. 3.2.2.3).

To achieve the desired virtual forces, we compute the required contact force in each foot. These contact forces are mapped into desired joint torques using the contact-point Jacobians. We then add compensation terms to the desired torques to account for gravitational and coriolis forces and send the resulting values to the low level torque controllers.

3.2.2.3 Phase Parametrization

In order to determine the timing of the transition from single to double stance and from double to single stance, we define a phase variable θ that takes on values from -1 to 1.

θ is given as a function of the horizontal distance from the body to the forward-most stance foot; this function is piecewise linear during each step and is reset to -1 at each touch-down. The function is determined by three parameters: x_{SS}, Δ_x, and l_{step}, as shown in Fig. 3.1a. l_{step} is the desired step length of the robot, x_{SS} is the distance that the body will cover during single stance, and Δ_x is the offset of the single stance phase from being centered around the stance foot. The net acceleration of the robot during single stance increases with the amount of time spent in front of the stance foot. Therefore, increasing/decreasing Δ_x leads to an increase/decrease in the average forward velocity.

3.2.2.4 Swing Foot Trajectory

During single stance, the swing foot trajectory is parametrized by θ. The trajectory is given as an elliptic arc which begins at the foot position after liftoff when $\theta = 0$

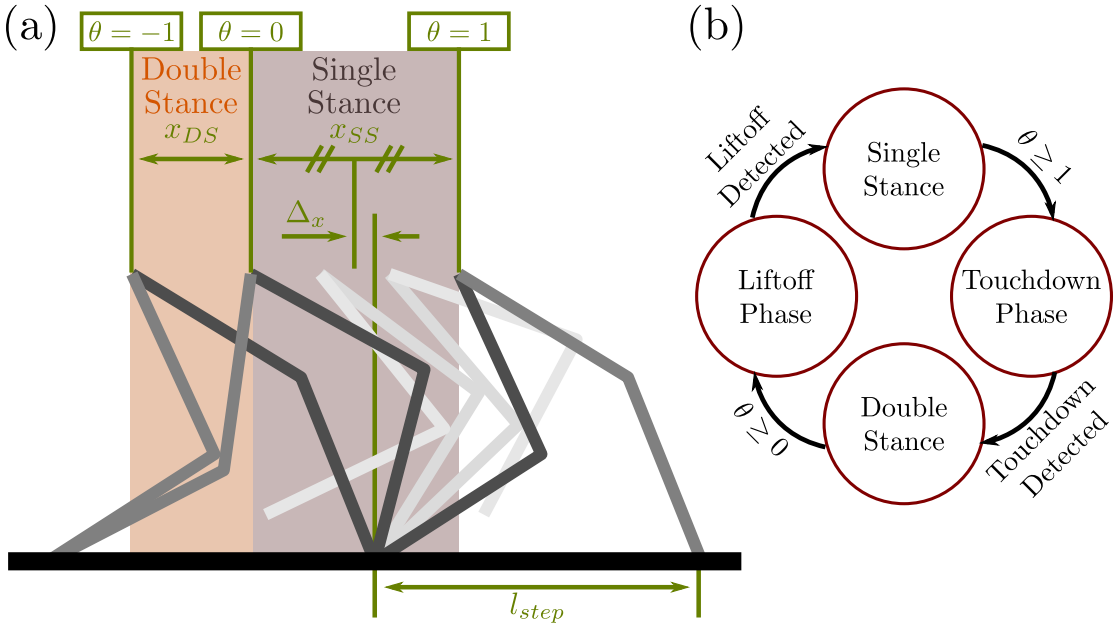

Figure 3.1: **(a)** Specification of the phase parameter θ. Between -1 and 0, and between 0 and 1, θ is a linear function of the horizontal distance from the forward-most stance foot to the main body. Note that if the gait is periodic with step length l_{step}, the duration of the double stance phase x_{DS} is determined by $x_{DS} = l_{step} - x_{SS}$. The parameter Δ_x (negative as shown in the figure) is used to control the average walking speed of the robot.
(b) Gait transition sequence. Beginning in double stance, θ becomes less negative as the body moves forwards, triggering the liftoff phase when it crosses 0. In this phase, the hip motor is held fixed and the knee motor retracts until contact is no longer sensed in the swing foot. Now in single stance, θ continues to increase from 0 as the body moves forward, until touchdown is initiated when θ crosses 1. Here the swing foot is held at fixed distance in front of the body until contact is sensed. θ is then reset to -1 and we enter double stance.

and ends on the ground at the desired step length when $\theta = 1$. This trajectory is recalculated after each liftoff event in order to ensure continuity of the desired swing-foot position with respect to time. The desired foot positions are mapped using inverse kinematics into desired joint positions which are then sent to the low level position controllers.

3.2.2.5 Contact Transitions

While implementing and tuning the controller on hardware, a consistent set of issues arose when switching between position control (in swing) and torque control (in stance) of the leg. When switching from double stance to single stance, delays in unloading the joints would lead to foot scuffing, as the foot would be moved forward along its swing trajectory before it had left the ground. When switching from single to double stance, the swing foot contacting the ground while in position control mode led to large disturbance forces. These difficulties are similar to challenges encountered when implementing contact-dependent controllers on other walking robots [42].

To alleviate the effect of this switch in controllers, we added two transition phases to the control of the robot, a lift-off phase when transitioning from double into single stance, and a touchdown phase when transitioning from single into double stance (see Fig. 3.1b). In the liftoff phase, the hip motor is held in position and the knee motor retracts until contact is no longer sensed. In the touchdown phase, the foot is held fixed with respect to the main body until contact is sensed.

3.2.3 Simulation

To facilitate development on the hardware, a realistic simulation of the robot was developed. The simulation includes models of the links, springs, motors, motor controllers, and planarizer, and uses a coulomb friction model of foot contact, with a friction coefficient of 0.5.

The rigid body model of the robot and the models of the series elastic elements are taken from [88]. The spring models include damping in the springs as well as the unilateral spring-damper in the knees. A mass-damper model of the motors and gearboxes was identified using data from the hardware, and the motor controller model was validated against the same data. Some effects of the planarizer were captured using Coulomb friction in the sliders and by adding mass to the x coordinate of the robot. The friction coefficients and mass properties were identified through hardware experiment.

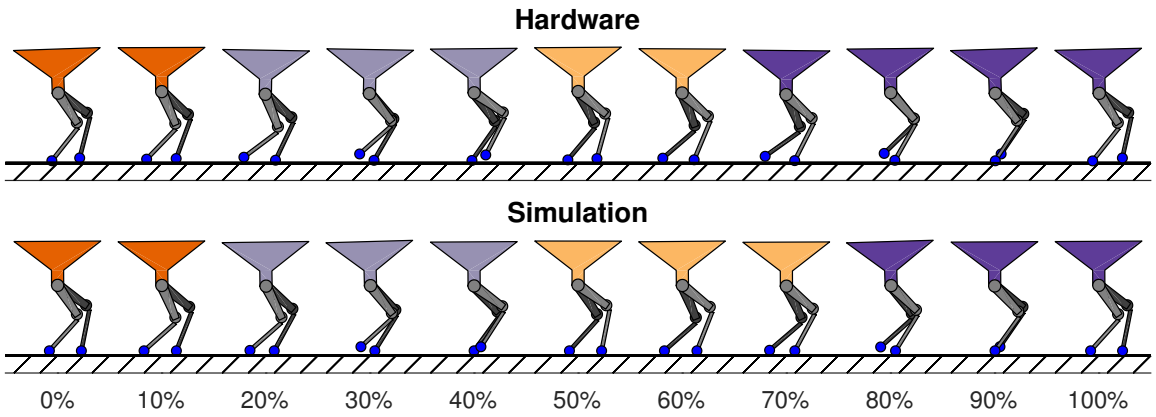

Figure 3.2: Stills of the robot across stride percentage in both hardware and simulation for a desired body height of 0.54 m and desired walking speed of 0.2 m/s.

3.3 Results

3.3.1 Walking Controller

3.3.1.1 Hardware Results

We achieved stable walking on the robot for desired heights ranging from 0.515 m to 0.57 m (81.7% to 90.4% of the straight-leg standing height of the robot) at a desired walking speed of 0.2 m/s (Froude number 0.0083), see Fig. 3.2. The parameter Δ_x was adjusted for each height but all other tuning parameters were fixed. At heights below 0.515 m, the controller had difficulty clearing the ground with the swing foot. At heights above 0.57 m, the knees became nearly straight at the end of double stance, leading to singularities in the controller and causing the robot to fall.

Fig. 3.3 shows the tracking of the control objectives during one stride of walking at a desired height of 0.54 m. The controller keeps the main-body height within about 2 % of the desired value throughout the stride. A small rise during the transition from single to double stance is caused by a sensing delay of ∼0.05 s in the foot contact sensors. Because we adjust the desired contact force in the hind foot to account for front foot forces only when contact is sensed, this delay causes the total normal contact force to temporarily exceed the weight of the robot.

The body pitch of the robot is controlled to within 0.04 rad (2.3 deg) from the vertical position. The largest pitch disturbances occur during left and right single stance and are asymmetric: during left single stance the robot leans backwards, while in right stance it leans forwards.

The main-body horizontal velocity shows significant variation, as it is not directly controlled. The body decelerates while its center of mass is behind the stance foot,

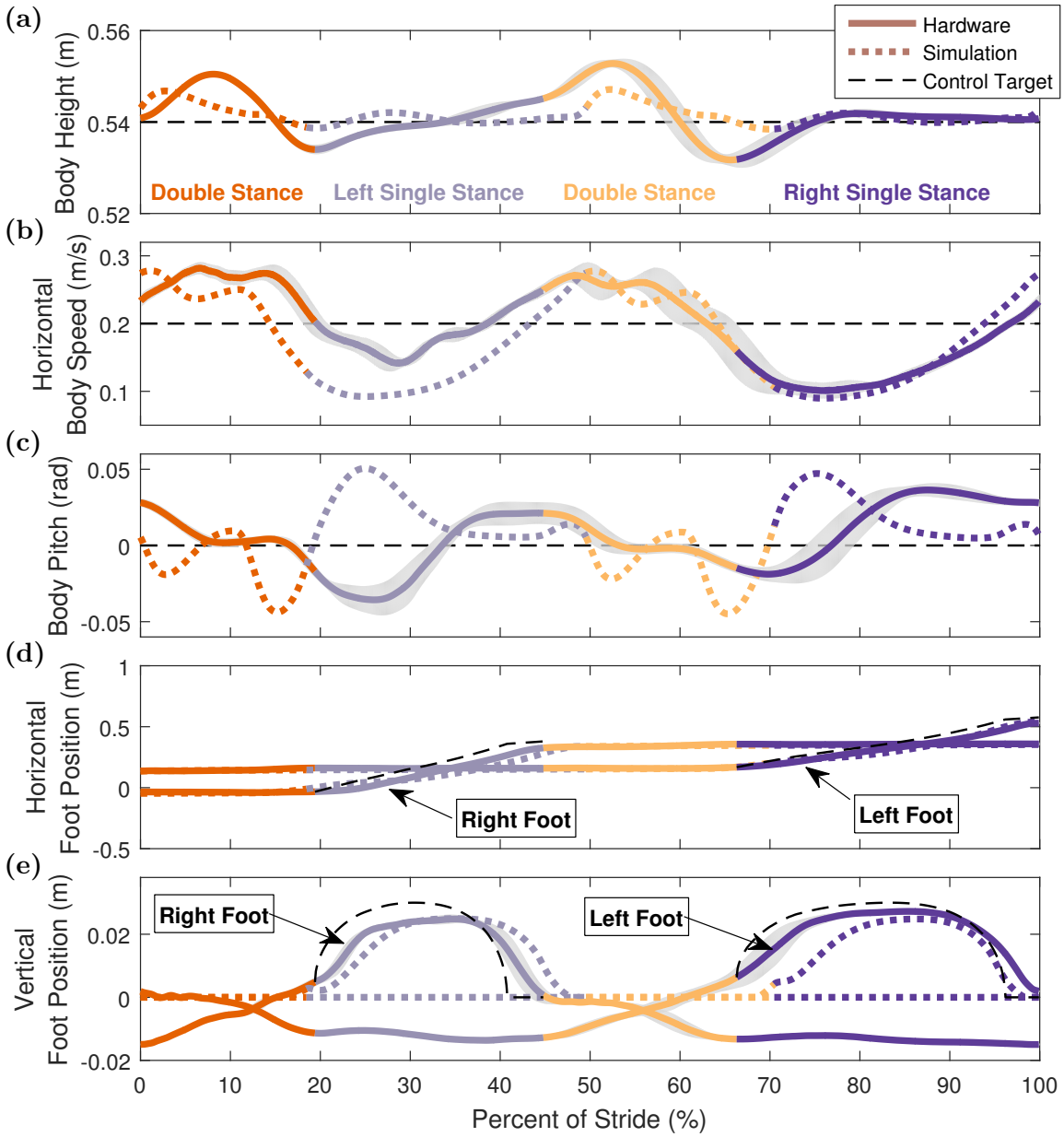

Figure 3.3: Tracking of controller objectives in simulation and hardware across stride percentage for a desired body height of 0.54 m and desired speed of 0.2 $^{m}/_{s}$. Plotted values are averaged over 30 strides, and one standard deviation is shaded in grey. The color of each line corresponds to the current stance configuration of the robot. In **(a)** we see that body height is maintained to within 5 mm of the desired value, with a slight rise upon entering double stance that occurs in both simulation and hardware. The horizontal body speed in **(b)** varies more significantly, as it is not controlled directly. The pitch in **(c)** is controlled to within 0.04 rad in hardware, with similar variation in simulation, however the phase and timing of the oscillations are different between the two. In **(d)** and **(e)**, we show the x and y coordinates of each foot with respect to the ground. During swing phase, the desired trajectory of each foot is shown as a thick dashed line.

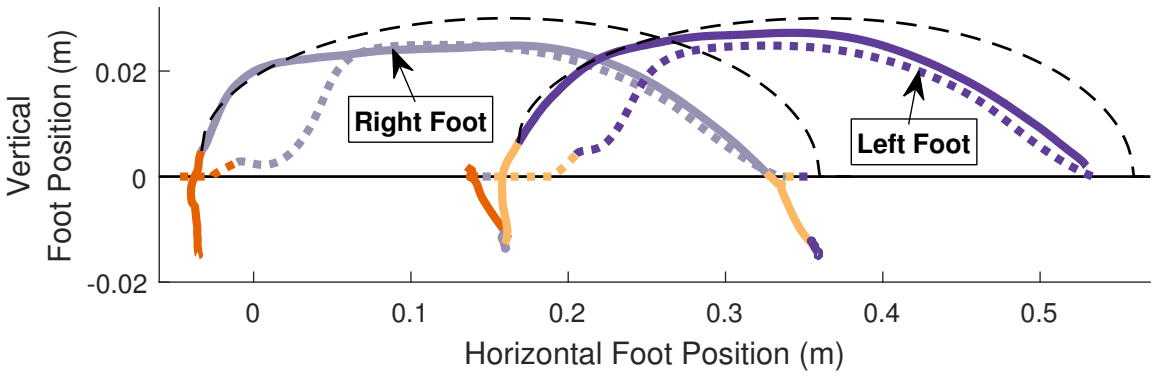

Figure 3.4: Trace of the foot trajectories with respect to the ground. We see significant deflection of the foot during stance phase (represented by negative y position). Also note that each foot slips slightly along the ground before liftoff in simulation, but not in hardware.

and accelerates when moving ahead of the stance foot. The horizontal velocity shows an asymmetry similar to the pitch: it diverges from the desired speed more during right single stance than in left single stance.

In Figs. 3.3d, 3.3e, and 3.4, we show the desired and measured foot trajectories of the robot. The effects of the soft feet are apparent here, as we see up to 1.4 cm of foot compression during single stance. Additionally, there is a small phase lag in the measured foot position behind its desired trajectory. Hence, the average stride length of the robot (0.365 m) is shorter than the desired value of $2l_{step} = 0.4$ m.

3.3.1.2 Simulation Results

To validate the accuracy of our model, an identical version of the walking controller was implemented in simulation. We found stable periodic gaits for each of the desired heights for which the physical robot walked. The simulated trajectories for the desired height of 0.54 m are shown as dotted lines in Figs. 3.3 and 3.4.

Both the main-body height and velocity trajectories in simulation largely agree with observations from hardware. The body height rises at the beginning of double stance, although with a somewhat smaller magnitude compared to hardware. The body velocity has a dip during single stance which is symmetric between left and right stance, in contrast to hardware. The pitch behavior in simulation is noticeably different from that in hardware: pitch oscillations have similar magnitudes, but differ in timing and direction. The left-right asymmetry observed in hardware does not occur for the pitch trajectories in simulation. We are uncertain of the cause of these discrepancies.

The simulated foot trajectories in Figs. 3.3d, 3.3e, and 3.4 show differences with those observed in hardware. The simulated feet don't penetrate the ground during stance and the rear foot slides along the ground near the end of double stance in simulation. Both of these effects can be attributed to inaccuracies in our foot contact model.

Finally, the stride period is longer in simulation (2.30 s) than in hardware (1.96 s) which results in the simulation having a lower average speed (0.174 m/s vs 0.196 m/s).

3.3.2 Cost of Transport

We characterize the energetic economy of gait in RAM*one* by computing the normalized electrical cost of transport or CoT [27]:

$$CoT = \frac{c}{mg\Delta x}, \tag{3.3}$$

where c is the electrical work used during a single stride, Δx is the distance traveled in the stride and mg is the total weight of the robot.

To compute the electrical work c, we integrate the instantaneous electrical power needed to drive the motors over the course of each stride. Also, because all motors on RAM*one* share an input voltage rail, electrical power generated from one motor can be directly consumed by the other motors. The total power is thus the sum of the mechanical power and the copper losses of all four motors i. Because the robot has no means of storing excess electrical energy, any negative net power generated by the motors is dissipated in the form of shunt losses. We thus ignore negative values of the instantaneous electrical power when integrating over a full stride with period t_F:

$$c = \int_0^{t_F} \max\left\{\sum_{i=1}^{4} T_i\dot{u}_i + \frac{T_i^2}{S_{mot}}, 0\right\} dt. \tag{3.4}$$

Here T_i and u_i are the on-board motor torque and velocity measurements of the EPOS3 motor controllers, and S_{mot} is the speed-torque gradient of the motors.

We computed the CoT from both the hardware and simulation data for a range of desired body heights at a walking speed of 0.2 m/s (see Fig. 3.5). Both in hardware and simulation, the CoT has a minimum at the height of 0.56 m. Below this height, the CoT increases linearly as the height decreases, with a slope of about 3 1/m.

This trend in CoT is consistent with previous results showing that compass-gait walking with straight legs (*i.e.* large body height) is the optimal motion at low speeds

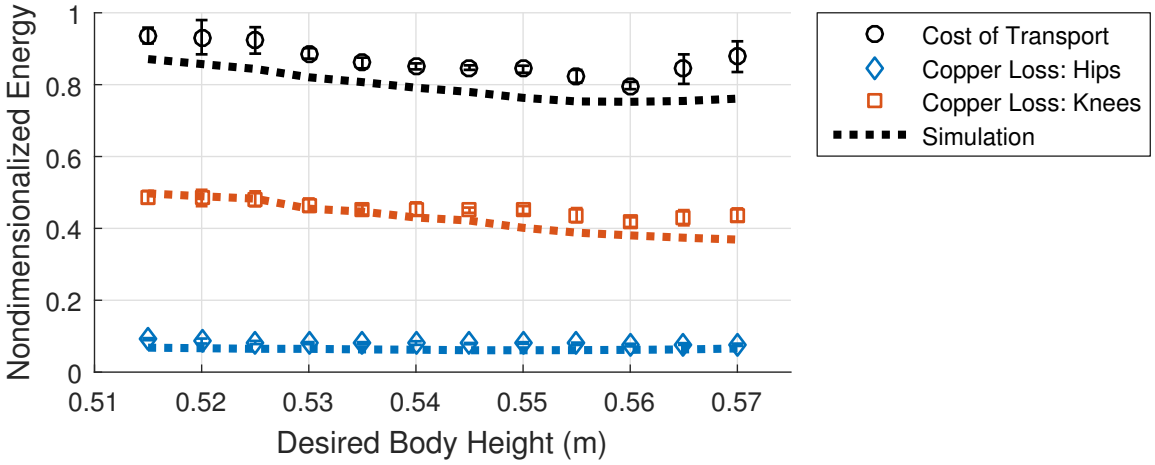

Figure 3.5: Cost of transport and nondimensionalized copper losses in the hips and knees across desired body height in hardware and simulation. A minimum in the CoT was observed in both simulation and hardware at a desired body heights of 0.56 m. Below this height, the CoT decreased linearly with increasing body height, with a slope of -2.835 $^1/_\mathrm{m}$ in hardware, and -3.08 $^1/_\mathrm{m}$ in simulation. The decrease can be attributed to the decreased knee copper losses incurred by a straighter legged gait. This is supported by the observation that on average, the knee copper losses decrease with walking height while the hip copper losses remain comparatively constant.

for a model of RAM*one* [88]. This is also similar to the energetic benefit of straight legs observed in human walking [32].

One explanation for the decrease in CoT with increasing heights is that straighter knees require less torque to support the weight of the robot, which in turn leads to lower copper losses in the knee motors. In Fig. 3.5, we plot normalized copper losses both for the knees and hips, for different walking heights. In agreement with our hypothesis, we see that the knee copper losses decrease for larger heights, while the hip losses remain about constant. This trend is observed both in simulation and hardware.

We also note that above the desired body height of 0.56 m, the CoT grows, until failure occurs above 0.57 m. For these larger body heights, the observed gaits, both in simulation and hardware, become irregular and unstable. Such instability is a result of straighter knees generating near-singular contact Jacobians during force control. Hence, the increase in CoT is likely due to greater control effort needed to stabilize these irregular gaits.

3.4 Discussion & Conclusion

This chapter introduced RAM*one*, a series elastic actuated planar bipedal robot designed to study the energetics of various gaits in bipedal locomotion. RAM*one*'s lightweight legs with high compliance series elastic joints allow for stable and economical locomotion. A walking controller based on virtual model control was designed and tested on hardware and in simulation.

Stable walking gaits were found in hardware for a walking speed of $0.2\,^{\mathrm{m}}/_{\mathrm{s}}$ over a range of desired body heights from 0.515 m to 0.57 m. The body height was tracked to within 2 % of the desired value and the pitch was kept within 0.04 rad (2.3 deg) of upright. The body velocity showed larger variation. Additional observations showed that these gaits were able to continue indefinitely and that the robot could walk over small obstacles.

The controller used in hardware was tested on a simulation of RAM*one*. Using the same tuning parameters from the robot, this controller led to stable walking gaits at each of the desired heights for which stable walking was observed in hardware. The qualitative behaviour of the simulation matched that of hardware, with similar trends occurring in the body height, pitch, horizontal body velocity and foot trajectories. Quantitatively, we found a cost of transport in simulation that was on average within 8.32 % of hardware measurements. The high fidelity of the simulation has proven to be valuable during the development process. For instance, we were able to use simulation to verify the hypothesis that delay in the foot contact sensor leads to the rise in main body height observed after touchdown in hardware.

Some differences remain between simulation and hardware, the most notable of which is the foot contact behaviour. Our current rigid contact model cannot account for the significant deformation that occurs in the real stance feet and fails to replicate the behaviour of the feet prior to liftoff.

Finally, we showed evidence that walking with straighter knees is energetically beneficial for this robot. This effect is likely due to the reduced electrical loss in the knee motors which results from needing smaller torques to support the robot. Even without explicitly creating a controller to minimize the cost of transport, we observed a minimum value of approximately 0.8 . This is comparable to the ~ 0.6 CoT of the robot ERNIE [109], which was achieved with an optimized controller.

The work here demonstrates the ability of RAM*one* to walk with a virtual model controller. Before performing more challenging studies, we plan to test the robot performance in several other basic tasks. These include: *(i)* walking steadily for a

range of forward speeds; *(ii)* walking under various disturbances; *(iii)* hopping in place. These tasks are stepping stones towards our long-term goal of economical and robust locomotion with the robot RAM*one*.

CHAPTER IV

Safety as a Constraint: Viability-based Control of Hybrid Systems

In the preceding chapter, we observed that desirable behaviour (economical walking) can often occur near the boundary of failure (straight knees leads to unstable behaviour). To better negotiate the failure boundary, this chapter aims to re-formulate the idea of avoiding falls as a constraint on the space of control inputs. By enforcing this constraint, robots will be free to conduct a wider range of behaviour without fear of falling. A regulator that enforces this constraint is demonstrated using a simple 4-dimensional compass-gait walking model.

4.1 Introduction

Failure can be catastrophic during the control of cyber-physical systems. Car crashes or robot falls are examples of costly events that should be avoided whenever possible. This is a challenging task for dynamical systems, which often contain states that unavoidably lead to failure, regardless of input. Take for example a moving car that is approaching an obstacle and moving too quickly to stop in time. In this case, the car has not yet failed (collided with the obstacle), but no control action can keep it safe.

Avoiding the set of all such states that inevitably fail (or the *unsafe set*) is a necessary and sufficient condition for preserving safety. With a direct representation of this unsafe set, a regulator can passively monitor the system state and take decisive action only when the state is at risk of entering the set. Such a regulator would guarantee safe operation, while allowing a secondary control system to remain flexible as long as safety is not threatened. An example of this is an automatic braking system, which acts if an accident is imminent, but otherwise does not impede the

driver. We refer to this as *semi-autonomous* control, since the dynamical system is autonomous when the regulator is active, but can be freely controlled when passive. However, exact computation of this unsafe set is a challenging problem for nonlinear and hybrid systems.

There are two broad approaches to computing the complement of the unsafe set, which we refer to as the *safe set*: those based on *reachability* and those based on *viability*. In reachability (or capturablity) analysis, safe sets are found by their ability to reach a known safe target set. For a walking robot, this could be reaching a stationary standing position [73, 71], or reaching a pre-computed periodic trajectory [87, 96].

The reachability definition of a safe set requires *a priori* knowledge of the desired safe target set. However, the appropriate target state is typically defined in a task specific manner. Consider for instance a legged robot. For level ground walking, a stationary standing state may be the most appropriate target set, since most safe motions can reach standstill. Unfortunately this is not necessarily true on a hill, where the robot may be unable to stop while safely running. If safety is defined based on the ability to come to a stationary position, this running motion would still be treated as unsafe, even though it would not produce a fall.

To avoid this unintended conservatism, this chapter explores a viability-based analysis, in which a set is safe if each state can avoid failure states for all time [8]. For the walking robot example, these failed states could be all states that fall or states that violate manually defined joint constraints [41, 65]. In legged robotics, viability-based computational approach for finding safe sets are often based on barrier functions [72, 41, 65]. These functions generate a boundary around failure sets, along with a guarantee that this boundary cannot be crossed.

One method to generate barrier functions for control systems uses a backstepping approach to enforce relative-degree n safety constraints under continuous dynamics [41, 66, 65]. However, in hybrid systems, discrete changes in the state are not accounted for. The state can thus escape the barrier function boundary at these points, which eliminates the viability guarantee. An example of this would be a walking robot that falls backwards due to energy lost through touchdown events. An alternative method, which guarantees barrier function viability through discrete events, is based on sums-of-squares programming [72]. This chapter extends these methods to enable simultaneous control synthesis to enable semi-autonomous safe control of hybrid systems.

Maintaining viability for a control system places a constraint on the control inputs.

The premise of semi-autonomous safe control is that this constraint must always be satisfied while other control objectives are approached. This trade-off is typically posed in the form of a quadratic program that must be solved in real-time [41, 6]. Though convex, these programs may still be computationally taxing and may not always have feasible solutions. This chapter presents an alternative method that allows for direct computation of the control input. This scheme takes an arbitrary user input, and modifies it only when viability is at risk. The resulting controller is guaranteed to preserve viability.

The contributions of this chapter are three-fold: first, a computational formulation for simultaneously generating hybrid barrier functions and safe controllers for hybrid control systems with uncertainty; second, a method for constructing a smooth guaranteed safe semi-autonomous hybrid controller; and, finally, an implementation and interactive demo of this guaranteed safe semi-autonomous controller on a compass gait walking model.

The rest of this chapter is organized as follows: Section 4.2 formally defines the two-part objective of this chapter. The first part (generating a viability certificate) is then numerically formulated using barrier functions in Section 4.3, and solved computationally using methods presented in Section 4.4. The resulting viability certificate is then used to generate a guaranteed safe semi-autonomous controller in Section 4.5. The overall method is implemented on two example systems in Section 4.6: a Dubins car and a compass gait walker.

4.2 Problem Statement

This section begins by formulating the problem that is solved in this chapter. We first define some notation: Let $\{A_i\}_{i\in\mathcal{I}}$ be a family of non-empty sets indexed by i, the *disjoint union* of this family is $\coprod_{i\in\mathcal{I}} A_i = \bigcup_{i\in\mathcal{I}}(A_i \times \{i\})$. The boundary of the set X is denoted ∂X. For a matrix A, $A \succeq 0$ represents A being positive semidefinite.

4.2.1 Dynamics

This chapter focuses on systems of the following form:

Definition IV.1. A *controlled hybrid system with bounded disturbance* is a tuple $\mathcal{H} = (\mathcal{I}, \mathcal{E}, \mathcal{X}, U, D, \mathcal{F}, \mathcal{S}, \mathcal{R})$, where:

- $\mathcal{I}$ is a finite set indexing the discrete states of $\mathcal{H}$;
- $\mathcal{E} \subset \mathcal{I} \times \mathcal{I}$ is a set of edges, forming a directed graph structure over $\mathcal{I}$;

- $\mathcal{X} = \coprod_{i \in \mathcal{I}} X_i$ is a disjoint union of domains, where each X_i is a compact subset of $\mathbb{R}^{n_i}$, and $n_i \in \mathbb{N}$;
- U is a compact subset of $\mathbb{R}^{m_u}$ that describes the range of control inputs, where $m_u \in \mathbb{N}$;
- D is a compact subset of $\mathbb{R}^{m_d}$ that describes the range of disturbance inputs, where $m_d \in \mathbb{N}$;
- $\mathcal{F} = \{F_i\}_{i \in \mathcal{I}}$ is the set of vector fields, where each $F_i : X_i \times U \times D \to \mathbb{R}^{n_i}$ is the vector field defining the dynamics of the system on X_i;
- $\mathcal{S} = \coprod_{e \in \mathcal{E}} S_e$ is a disjoint union of guards, where each $S_{(i,i')} \subset \partial X_i$ is a compact, co-dimension 1 guard (with respect to X_i), defining a state-dependent transition to $X_{i'}$;
- $\mathcal{R} = \{R_e\}_{e \in \mathcal{E}}$ is a set of reset maps, where each map $R_{(i,i')} \colon S_{(i,i')} \to X_{i'}$ defines the transition from guard $S_{(i,i')}$ to $X_{i'}$.

We make the following assumptions about these systems:

Assumption 1. Each vector field $F_i \in \mathcal{F}$ is affine in control and disturbance. That is: $F_i(x, u, d) = f(x) + g_u(x)u + g_d(x)d$. Where $f : X_i \to \mathbb{R}^{n_i}$, $g_u : X_i \to \mathbb{R}^{n_i \times m_u}$ and $g_d : X_i \to \mathbb{R}^{n_i \times m_d}$ are Lipschitz functions.

Assumption 2. $U = [-1, 1]^{m_u}$, and $D = [-1, 1]^{m_d}$.

The first assumption ensures existence and uniqueness of solutions while the second assumption comes with limited loss of generality, as non-uniform control input and disturbance constraints can be realized by scaling the functions g_u and g_d, respectively. Next, we define *Lie Derivatives*:

Definition IV.2. Given a differentiable function $v : X_i \to \mathbb{R}$, the *Lie Derivatives*: $\mathcal{L}_f$, $\mathcal{L}_{g_u}$, and $\mathcal{L}_{g_d}$ are linear functions of v that are defined as follows: $\mathcal{L}_f v = \sum_{i=1:n} f^{(i)}(\cdot)\frac{\partial v(\cdot)}{\partial x^{(i)}}$, $\mathcal{L}_{g_u} v = \sum_{i=1:n} g_u^{(i)}(\cdot)\frac{\partial v(\cdot)}{\partial x^{(i)}}$, $\mathcal{L}_{g_d} v = \sum_{i=1:n} g_d^{(i)}(\cdot)\frac{\partial v(\cdot)}{\partial x^{(i)}}$, where $x^{(i)} \in \mathbb{R}$ is the $i'th$ element of x, $f^{(i)} : X \to \mathbb{R}$ is the $i'th$ element of f, $g_u^{(i)} : X \to \mathbb{R}^{1 \times n_u}$ is the $i'th$ row of g_u and $g_d^{(i)} : X \to \mathbb{R}^{1 \times n_d}$ is the $i'th$ row of g_d

Next, we define an *execution* of a hybrid system via construction in Algorithm 1 [15]. Step 1 initializes the execution at a given point (x_0, i) at time $t = 0$. Step 3 defines ϕ to be the maximal integral curve of F_i under the control u beginning from the initial point. Step 4 defines the execution on a finite interval as the curve ϕ with

Require: $t = 0$, $\tau > 0$, $i \in \mathcal{I}$, $(x_0, i) \in \mathcal{X}$, $u : \mathbb{R} \to U$, and $d : \mathbb{R} \to D$, with u, d Lebesgue measurable.

1: Set $x(0) = (x_0, i)$.
2: **loop**
3: Let $I \subset [t, \tau]$ be an interval and $\phi : I \to X_i$ be an absolutely continuous function such that:

i $\dot{\phi}(s) = F_i(\phi(s), u(s), d(s))$ for almost every $s \in I$ with respect to the Lebesgue measure on I with $(\phi(t), i) = x(t)$ and

ii for any other $\hat{I}$ and $\hat{\phi} : \hat{I} \to X_i$ satisfying (i), $\hat{I} \subset I$.

4: Let $t' = \sup I$ and $x(s) = (\phi(s), i)$ for each $s \in [t, t')$.
5: **if** $t' = \tau$, **or** $\nexists (i, i') \in \mathcal{E}$ such that $\phi(t') \in S_{(i,i')}$ **then**
6: Stop.
7: **end if**
8: Let $(i, i') \in \mathcal{E}$ be such that $\phi(t') \in S_{(i,i')}$.
9: Set $x(t') = R_{(i,i')}(\phi(t'))$, $t = t'$, and $i = i'$.
10: **end loop**

Algorithm 1: Execution of Hybrid System $\mathcal{H}$

associated index i. As described in Steps 5 - 7, the trajectory terminates when it either reaches the terminal time τ or hits $\partial X_i \setminus \bigcup_{(i,i') \in \mathcal{E}} S_{(i,i')}$ where no transition is defined. If the latter is true, we say that failure has occurred. Steps 8 and 9 define a discrete transition to a new domain using a reset map where evolution continues again as a classical dynamical system (Step 3).

Hybrid systems can suffer from Zeno executions, i.e. executions that undergo an infinite number of discrete transitions in a finite amount of time. We do not consider systems with Zeno executions, since the state of the trajectory may not be well defined after Zeno [7]:

Assumption 3. $\mathcal{H}$ has no Zeno execution.

4.2.2 Safety

In this chapter, *safety* is defined as being able to choose a controller which ensures that the continuous state of the hybrid system either stays within the interior of $\mathcal{X}$ or transitions to another discrete state. To understand this definition, consider a walking robot. The domain of the system can be defined as all kinematic configurations with only feet touching the ground with hybrid transitions occurring when a foot impacts the ground. Failure for this system would correspond to when the robot reaches the edge of the domain without intersecting a guard, e.g. when something other than the

foot hits the ground. This set of non-guard domain boundaries is referred to as the *Failure Domain*.

Definition IV.3. The *Failure Domain* or $\mathcal{X}^{\mathcal{F}}$ is defined as the disjoint union of *Failure Sets* $\coprod_{i\in\mathcal{I}} X_i^F$ where $X_i^F = \partial X_i \backslash \bigcup_{(i,i')\in\mathcal{E}} S_{(i,i')}$.

To guarantee safety, this chapter finds a *Viability Domain* [8]:

Definition IV.4. A *Viability Domain* $\mathcal{V} = \coprod_{i\in\mathcal{I}} V_i$ is a disjoint union of domains $V_i \subset X_i$, $V_i \bigcap X_i^F = \emptyset$, which is forward control invariant. That is, there exists a state feedback controller $\coprod_{i\in\mathcal{I}} u_i$ with $u_i : X_i \to U$ Lipschitz, such that for every initial condition $(x_0, i) \in \mathcal{V}$, and for every time varying disturbance signal $d : \mathbb{R}^+ : D$, the execution of the system from the initial condition remains in $\mathcal{V}$ for all time $\tau \in [0, \infty)$. We refer to any feedback controller that is able to ensure that the system is forward control invariant as an *Autonomous Viable Controller*.

Note that in this definition the control input relies on state feedback while the disturbance input d is a time-varying signal. The feedback ensures that control input is able to adapt to the effects of disturbance without having direct access to it. The forward control invariance property ensures that any state that begins within a viability domain $\mathcal{V}$ can be controlled to remain within the domain. Since $\mathcal{V}$ is strictly contained within the system domain, we know that our safety criteria can be maintained by at least one controller for all states within $\mathcal{V}$. Therefore we also refer to viability domains as "safe sets."

4.2.3 Semi-Autonomous Safe Control

In constructing a controller that maintains viability, we draw inspiration from the *Inertia Principle* [8], which states that biological macrosystems maintain constant inputs as long as viability is not at stake. This principle suggests the form of a semi-autonomous, safety preserving controller: given an initial state within a viability domain, a user defined control input is applied without modification to the system; the state of the system is then continuously monitored and the control input is replaced by an autonomous viable controller if the state approaches the boundary of the viability domain. This regulation scheme is conceptually similar to the reference governor approach [9], in which a reference input is tracked unless safety is at risk.

4.2.4 Goal

Using these definitions, we state our objective as:

1. Find a viability domain and a corresponding autonomous viable controller. This can be stated mathematically as the following optimization problem:

$$\begin{aligned} &\sup_{V_i, u_i, \forall i \in \mathcal{I}} \sum_{i \in \mathcal{I}} \int_{V_i} dx \\ &\text{s.t. } \mathcal{V} = \coprod_{i \in \mathcal{I}} V_i \text{ is a Viability Domain} \\ &\qquad \mathcal{U} = \coprod_{i \in \mathcal{I}} u_i \text{ is a Viable Controller} \end{aligned} \tag{4.1}$$

   This problem is translated into a numerical representation in Section 4.3, and is then solved in Section 4.4.

2. Use this domain and autonomous viable controller to construct a semi-autonomous viable controller. To address this problem, we propose an interpolation scheme in Section 4.5.

4.3 Numerical Formulation

To represent Problem 4.1 numerically, we use barrier functions as has been done for hybrid systems in the literature [72]. Specifically, we represent each V_i as the zero super-level set of a differentiable function $v_i : X_i \to \mathbb{R}$:

$$V_i = \{x \mid v_i(x) \geq 0\}, \tag{4.2}$$

and we represent our autonomous viable controller using the Lipschitz functions $u_i : X_i \to \mathbb{R}^{n_u}$. We next describe how to translate the objective function and constraints in (4.1) into numerical constraints on v_i.

4.3.1 Constraints

Our first constraint ensures that: $V_i \bigcap X_i^F = \emptyset$ (this is a necessary condition for $\mathcal{V}$ to be a Viability Domain). To enforce this, we require the functions v_i to satisfy the following condition:

$$v_i(x) < 0, \quad \forall x \in X_i^F \tag{4.3}$$

The next constraint we enforce is the control input bounds:

$$-1 \leq u_i(x) \leq 1 \quad \forall x \in X_i \tag{4.4}$$

Given these constraints, a sufficient requirement to ensure that $\mathcal{V} = \coprod_{i \in \mathcal{I}} V_i$ is a Viability Domain is satisfying the following two conditions:

Definition IV.5. The *Continuous Invariance Condition* for each $i \in \mathcal{I}$ is

$$\mathcal{L}_f v_i(x) + \mathcal{L}_{g_u} v_i(x) u_i(x) + \mathcal{L}_{g_d} v_i(x) d > 0 \tag{4.5}$$

for all $x \in \partial V_i$ and for all $d \in D$ and ensures that the system state remains within the Viability Domain during continuous evolution.

Definition IV.6. The *Discrete Invariance Condition* is:

$$v_{i'}(R_{(i,i')}(x)) \geq v_i(x) \quad \forall x \in S_{(i,i')}, \tag{4.6}$$

which ensures that the system state remains within the Viability Domain during discrete updates.

Theorem 1. The disjoint union of the zero super-level sets of any collection of differentiable functions that satisfy the Continuous and Discrete Invariance Conditions is a Viability Domain.

Proof. Given a point x_0 in mode i that satisfies $v_i(x_0) > 0$, suppose there exists an execution time τ, mode j, and disturbance d such that the state of the hybrid system under the feedback control beginning from x_0 and under the disturbance d arrives at a state $x_f \in X_j^F$ at time τ. As a result of (4.3), $v_j(x_f) < 0$.

Since the dynamics are Lipschitz and the barrier function is differentiable, the value of the barrier function is discontinuous in time only when the execution passes through a guard. (4.6) implies that the barrier function value can only increase at these points, therefore we know that the value must cross zero in a continuous domain, say mode k. Since the value of v_k varies continuously in time at the moment of zero-crossing, we know $\dot{v} \leq 0$. However, this contradicts (4.5). Therefore no such execution exists, and the zero super-levelset of the barrier function is a viability domain. □

4.3.2 Objective Function

The objective function in (4.1) is difficult to compute exactly for an arbitrary polynomial barrier function, since the domain of integration is given by a semi-algebraic set. We propose an analytically tractable approximation to this objective:

$$\sum_{i \in \mathcal{I}} \int_{X_i} v_i(x) dx, \tag{4.7}$$

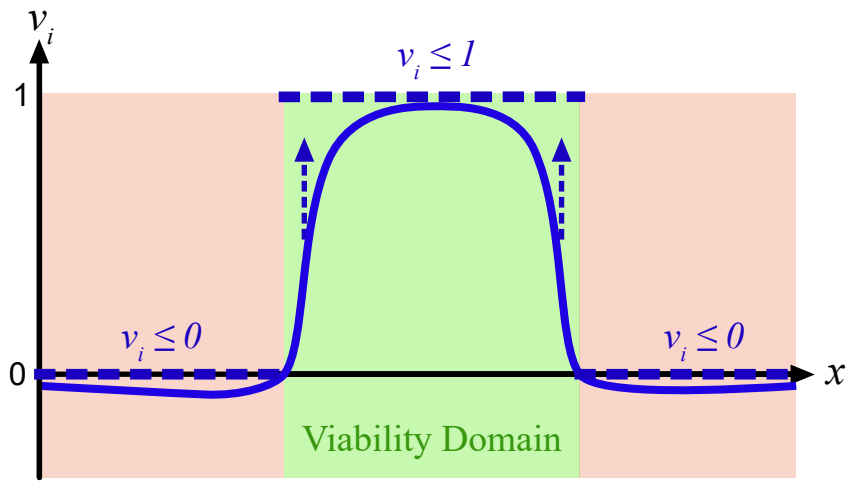

Figure 4.1: An illustration of how the objective function (4.7) approximates the volume of the viability domain. Take a set of differentiable functions v_i that satisfy the constraints of the previous section. For every point x not in the set ∂V_i, the value $v_i(x)$ is constrained only by (4.3) and $v_i(x) \leq 1, \forall x \in X_i$. This means that $v_i(x)$ can increase to a value of 1 for points inside the set (x s.t. $v_i(x) > 0$), and $v_i(x)$ increases to a value of 0 for points outside this set. As a result, each v_i approaches the indicator function over V_i, and the integral in Eq. 4.7 approaches the original objective.

along with the constraint $v_i(x) \leq 1, \forall x \in X_i$. Figure 4.1 illustrates how these constraints approximate the cost function in (4.1).

4.3.3 Numerical Optimization Problem

Combining these conditions, we can construct the following numerical optimization problem over the space of differentiable functions to represent (4.1):

$$\sup_{v_i, u_i \, \forall i, i' \in \mathcal{I}} \sum_{i \in \mathcal{I}} \int_{X_i} v_i(x) dx \tag{4.8}$$

$$\begin{aligned}
\text{s.t. } & v_i(x) \leq 1 && \forall x \in X_i \\
& v_i(x) \leq 0 && \forall x \in \partial X_i \\
& \mathcal{L}_f v_i(x) + \mathcal{L}_{g_u} v_i(x) u_i(x) + && \forall x \in \partial V_i, \\
& \qquad \mathcal{L}_{g_d} v_i(x) d \geq 0 && \forall d \in D \\
& -1 \leq u_i(x) \leq 1 && \forall x \in X_i \\
& v_{i'}(R_{(i,i')}(x)) \geq v_i(x) && \forall x \in S_{(i,i')}
\end{aligned}$$

4.4 Computational Implementation

This section describes a numerical approach to solve (4.8).

4.4.1 Function Representation with Polynomials

To represent the space of differentiable functions, we choose a polynomial basis set for the decision variables of (4.8) and additionally assume that the dynamics, reset maps, domains, and guards can all be represented with polynomials for each $i \in \mathcal{I}$:

Assumption 4. v_i, u_i are polynomial in state.

Assumption 5. f, g_u, g_d, and $R_{(i,i')}$ are polynomial in state.

Assumption 6. X_i and $S_{(i,i')}$ are semi-algebraic sets. That is there exist polynomial functions $h^j_{X_i}, h^j_{S_{(i,i')}} : X_i \to \mathbb{R}$ such that: $X_i = \left\{x \in \mathbb{R}^{n_i} \mid h^j_{X_i}(x) \geq 0, \forall j\right\}$ and $S_{(i,i')} = \left\{x \in \mathbb{R}^{n_i} \mid h^j_{S_{(i,i')}}(x) \geq 0, \forall j\right\}$.

4.4.2 Sums-of-Squares

The problem of finding polynomials that satisfy inequality constraints over sets can be solved using a sums-of-squares approach [70, 53], which we briefly summarize in this subsection. This approach replaces the constraint: $p \geq 0$ with the requirement that p is a sum-of-squares polynomial, i.e. $p = \sum_i p_i^2$ where p_i are polynomials. We refer to this condition as $p \in SoS$. The constraint $p \in SoS$ can be represented in matrix form as:

$$\begin{aligned} p &= m^T A m \\ A &\succeq 0 \end{aligned} \tag{4.9}$$

Where m is a vector of monomials in x, and A is a matrix of scalars. An optimization problem with constraints of this form and with a cost function that is a linear function of the coefficients of a polynomial can be represented as a Semi-Definite Program (SDP). These are a well-studied class of convex problems [14], which can be solved to global optimality with a range of commercial solvers.

The constraint $p \in SoS$ ensures that p is globally positive, i.e. $p(x) \geq 0, \forall x \in \mathbb{R}^{n_i}$. For the purpose of our problem, we instead enforce a local version of this constraint: $p(x) \geq 0, \forall x \in K$, with $K = \{x | h(x) \geq 0\}$. To do this, we introduce a slack polynomial s that is added to the decision variables to generate the modified constraint: $p - hs \in SoS$ and $s \in SoS$. One can show that under this constraint, the local positivity requirement is satisfied [95, 53].

4.4.3 Sufficient Invariance Conditions

Scaling is a major challenge in the sums-of-squares approach. The vector of monomials m in (4.9) scales combinatorially with the dimension of the constraint space and with the maximum polynomial degree [70]. This is particularly problematic in the *Continuous Invariance Condition* ((4.5)), which is enforced over the space $\partial V_i \times D$ This space has dimension $n_i + m_d$, which is larger than that of the remaining constraints (dimension n_i). To alleviate this, we construct a replacement condition of dimension n_i:

Definition IV.7. The *Sufficient Continuous Invariance Condition* for each $i \in \mathcal{I}$ is:

$$\begin{aligned} &\mathcal{L}_f v_i(x) + \mathcal{L}_{g_u} v_i(x) u_i(x) - \sum_{j \in \mathcal{J}_i} q_{ij}(x) + \lambda_i(x) v_i(x) \geq 0 \quad \forall x \in X_i \\ &q_i(x) - \mathcal{L}_{g_d} v_i(x) \geq 0 \qquad \forall x \in X_i \\ &q_i(x) + \mathcal{L}_{g_d} v_i(x) \geq 0 \qquad \forall x \in X_i \end{aligned} \tag{4.10}$$

where each λ_i is a polynomial, each q_i is an m_d dimensional vector of polynomials, and the last two lines are element-wise inequalities.

This reduced condition accomplishes two objectives: first it introduces the decision variable λ_i, which allows us to enforce the constraint over the whole set X_i instead of over ∂V_i (λ_i can slacken the constraint whenever $v_i \neq 0$); and second, the reduced condition places an upper bound on the effect of the disturbance using the functions q_i. This is formalized in the following theorem:

Theorem 2. If for each $i \in \mathcal{I}$, v_i, λ_i, u_i, and q_i meet the sufficient continuous invariance condition, then v_i and u_i satisfy the continuous invariance condition.

Proof. Take any mode i and point $x \in \partial V_i$ (i.e. $v_i(x) = 0$), and take any $d \in D$. Then the λ_i term drops out of the first condition in (4.10) giving us:

$$\mathcal{L}_f v_i(x) + \mathcal{L}_{g_u} v_i(x) u_i(x) - \sum_{j \in \mathcal{J}_i} q_{ij}(x) \geq 0 \tag{4.11}$$

Now since $D = [-1, 1]^{m_d}$, we know $\mathcal{L}_{g_d} v_i(x) d \geq -\sum |\mathcal{L}_{g_d} v_i(x)|$, where the absolute value is performed element-wise, and the sum is over all vector elements. Since the last two constraints of (4.10) imply that $|\mathcal{L}_{g_d} v_i(x)| < q_i(x)$ we know $\mathcal{L}_{g_d} v_i(x) d > -\sum_{j \in \mathcal{J}_i} q_{ij}(x)$ giving us $\dot{v}_i(x, u_i(x), d) > 0$. □

4.4.4 Bilinear Sums-of-Squares Problem

We replace the positivity constraints in (4.8) with sums-of-squares constraints, and replace the *Continuous Invariance Condition* with the sufficient condition from Section 4.4.3, which results in the following optimization problem:

$$\sup_{\substack{v_i,u_i,\lambda_i,q_i \\ \sigma_{1_i},\ldots,\sigma_{9_i}\ \forall i\in\mathcal{I}}} \sum_{i\in\mathcal{I}} \int_{X_i} v_i(x)dx \tag{4.12}$$

$$\begin{aligned}
\text{s.t. } & 1 - v_i - h_{X_i}\sigma_{1_i} && \in SoS \\
& - v_i + h_{X_i}\sigma_{2_i} - h_{X_i}\sigma_{3_i} && \in SoS \\
& \mathcal{L}_f v_i + \mathcal{L}_{g_u} v_i u_i + \sum_{j\in\mathcal{J}_i} q_{ij} + \lambda_i v_i - h_{X_i}\sigma_{4_i} && \in SoS \\
& q_i - \mathcal{L}_{g_d} v_i - h_{X_i}\sigma_{5_i} && \in SoS \\
& - q_i(x) + \mathcal{L}_{g_d} v_i - h_{X_i}\sigma_{6_i} && \in SoS \\
& u_i(x) + 1 - h_{X_i}\sigma_{7_i} && \in SoS \\
& - u_i(x) + 1 - h_{X_i}\sigma_{8_i} && \in SoS \\
& v_{i'}(R_{(i,i')}(x)) - v_i(x) - h_{S_{(i,i')}}\sigma_{9_i} && \in SoS \\
& \sigma_{1_i},\ldots,\sigma_{9_i} && \in SoS
\end{aligned}$$

Here σ_{j_i} are vectors of polynomials in x_i described in Section 4.4.2. To express a sum-of-squares problem as an SDP, all SoS constraints must be linear functions of the decision variable polynomials. However, the above problem includes the terms $\mathcal{L}_{g_u} v_i u_i$ and $\lambda_i v_i$ which are *bilinear* in u_i, v_i and in λ_i, v_i respectively. Problems of this form are referred to as bilinear sums-of-squares problems. The bilinear nature of the constraints means that these problems are non-convex, and we can no longer guarantee a globally optimal solution to this problem. We use an alternation approach to solve this problem as described in the next section.

4.4.5 Alternation

To solve the nonconvex bilinear sums-of-squares program (4.12) we solve a pair of convex optimization SDPs. In each program one of the bilinear variables is kept fixed while the other is optimized over. The variables that are optimized are then fixed while the other pair of variables are optimized. If the final solution satisfies the constraints of the original program, the solution is guaranteed to be a Viability

Domain. Computationally, each SDP is formulated in spotless[1] and solved using Mosek.

4.4.5.1 Alternation Steps

We arrive at the alternation steps by splitting the non s-function decision variables from (4.12) into two groups: (v_i, q_i) and (u_i, λ_i). We use a separate set of s-functions as decision variables for each step. Keeping the second group of decision variables fixed, we arrive at the first convex sums-of-squares alternation step:

$$\begin{aligned}
&\sup_{\substack{v_i,u_i,\lambda_i,q_i,\\ \sigma_{a1_i},\ldots,\sigma_{a7_i}\ \forall i\in\mathcal{I}}} \sum_{i\in\mathcal{I}} \int_{X_i} v_i(x)dx && && (4.13)\\
&\text{s.t. } 1 - v_i - h_{X_i}\sigma_{a1_i} && \in SoS\\
&\quad - v_i + h_{X_i}\sigma_{a2_i} - h_{X_i}\sigma_{a3_i} && \in SoS\\
&\quad \mathcal{L}_f v_i + \mathcal{L}_{g_u} v_i u_i + \sum_{j\in\mathcal{J}_i} q_{ij} + \lambda_i v_i - h_{X_i}\sigma_{a4_i} + \max(c,0) && \in SoS\\
&\quad q_i - \mathcal{L}_{g_d} v_i - h_{X_i}\sigma_{a5_i} && \in SoS\\
&\quad - q_i(x) + \mathcal{L}_{g_d} v_i - h_{X_i}\sigma_{a6_i} && \in SoS\\
&\quad v_{i'}(R_{(i,i')}(x)) - v_i(x) - h_{S_{(i,i')}}\sigma_{a7_i} && \in SoS\\
&\quad \sigma_{a1_i},\ldots,\sigma_{a7_i} && \in SoS
\end{aligned}$$

Keeping the first group fixed, the second convex step is:

$$\begin{aligned}
&\inf_{\substack{c,u_i,\lambda_i,\sigma_{b1_i},\\ \sigma_{b2_i},\sigma_{b3_i}\ \forall i\in\mathcal{I}}} c && && (4.14)\\
&\text{s.t. } \mathcal{L}_f v_i + \mathcal{L}_{g_u} v_i u_i + \sum_{j\in\mathcal{J}_i} q_{ij} + \lambda_i v_i - h_{X_i}\sigma_{b1_i} + c && \in SoS\\
&\quad u_i(x) + 1 - h_{X_i}\sigma_{b2_i} && \in SoS\\
&\quad - u_i(x) + 1 - h_{X_i}\sigma_{b3_i} && \in SoS\\
&\quad \sigma_{b1_i}, \sigma_{b2_i}, \sigma_{b3_i} && \in SoS
\end{aligned}$$

In each of these programs, the variable c is added to the decision variables from (4.12) and is a slack variable that is used as an objective function in the second alternation step. This choice of objective function in the second step encourages u_i and λ_i to relax the invariance constraint of the first step. Provided the satisfaction of the

[1] `https://github.com/spot-toolbox/spotless`

additional condition $c \leq 0$, any variables $(v_i, q_i), (u_i, \lambda_i)$ that satisfy the constraints of the alternation steps, also satisfy the constraints of (4.12).

4.4.5.2 Objective Function Augmentation

When using low order polynomials, the objective function (4.7) may be a poor representation of the volume of the viable domain. For small viable domains, the negative value of v in the unsafe regions exceeds the positive values in the safe set. When the value of the integral becomes negative, it becomes optimal to scale the value of v down to 0, and a safe set may not be found. This is especially problematic in early steps of the optimization, before a large safe set is found.

To assist in conditioning during these early steps, we add a regularizing term to the objective function of (4.13):

$$\sum_{i \in \mathcal{I}} \int_{X_i} v_i(x) dx + w_{tp} \sum_j v_i(tp_j) \tag{4.15}$$

Where tp_j are user chosen points that are known to be interior to the safe set, and w_{tp} is a weighting term. We call these points "tent-poles" since they bias the optimization to lift the value of v at known safe points. The weight of the tent-poles can be decreased as the optimization progresses, and the safe set becomes better represented.

4.4.5.3 Initialization, Alternation, and Convergence

To begin the alternation, the user must specify an initial choice of (u_i, λ_i, c). In practice, the most important tuning parameter of the optimization is the initial value of c. Choosing a value larger than 0 relaxes the continuous invariance condition, allowing for a non-empty Viability Domain to be found on the first step of the alternation. Any pair of solutions to the alternation steps with $c \leq 0$ is guaranteed to satisfy the constraints of (4.8). The form of the alternation ensures the value of c will not increase between steps. In practice if the initial value of c is too large, the alternation stalls before c crosses 0, and no feasible solution is found.

Once the value of c is reduced below 0 in the alternation steps, subsequent steps are guaranteed to not decrease the objective function of the first alternation. We terminate the alternation once the relative increase decreases below a user-specified threshold. Since the objective function is bounded from above by $\sum_{i \in \mathcal{I}} \int_{X_i} dx + n_{tp} w_{tp}$, the alternation is guaranteed to converge (once a solution with $c \leq 0$ is found).

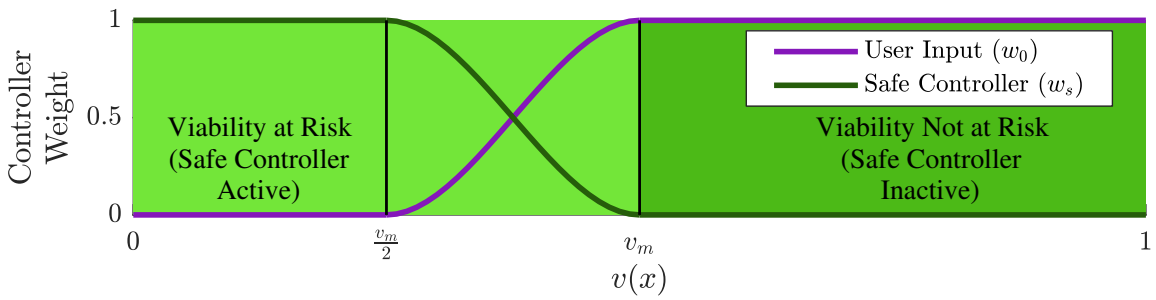

Figure 4.2: Scaling weights between the user input u_0 and the guaranteed safe controller u. The weights satisfy $w_0 + w_s = 1$ and are used to form the semi-autonomous controller $u_m = w_0 u_0 + w_s u$. When the barrier function value is above a threshold value (i.e. $v_i(x) > v_m$), the user input is unmodified, as we are sufficiently removed from the boundary of the safe set. When $0 \leq v_i(x) \leq \frac{v_m}{2}$, the safe controller is fully active, keeping the state in the safe set. Between these regions, the controller is smoothly interpolated with a cubic spline to ensure continuity of the semi-autonomous controller.

4.5 Guaranteed Safe Semi-autonomous Controller

We use a feasible solution to the bilinear sums-of-squares (4.12) to generate a guaranteed safe semi-autonomous controller. This controller modifies user input to ensure that the constraints of (4.1) are always satisfied. The two constraints that apply to the input u_i for $\mathcal{V}$ to be a viability domain are the input bounds (4.4) and the continuous invariance condition (4.5). Assuming that the user inputs are saturated to always satisfy the input bounds, this leaves the continuous invariance condition. Note that this condition is only active on the set ∂V_i, that is when $v_i(x) = 0$. This means that any controller is safe so long as it enforces the invariance condition in a neighborhood of $v(x) = 0$.

Since a controller that is discontinuous on the boundary of the safe set would pose difficulties for systems with finite bandwidth, we additionally must ensure that the new controller is continuous near the boundary. To achieve this, we define a mask that smoothly interpolates between the user input and the guaranteed safe controller u_i, which we know satisfies the safety condition on ∂V_i (Fig. 4.2).

4.6 Results

The method described in Sections 4.4 and 5.4.3 is applied to generate safe controllers for two systems: a Dubins car and a compass gait walker. The two-state Dubins car is used as a test case to verify and visualize the accuracy of the safety guarantee. The four-state hybrid dynamics of the compass gait walker present a more

complex example.

4.6.1 Dubins Car

We begin with a straight line lane-keeping task for a Dubins car model. The state consists of the lateral position in the lane (y) and car angle (θ). The domain is defined as: $X = \left\{(y, \theta) \mid y \in [-0.5, 0.5],\, \theta \in [-\frac{\pi}{4}, \frac{\pi}{4}]\right\}$. The dynamics of the model are given by:

$$\begin{bmatrix} \dot{y} \\ \theta \end{bmatrix} = \begin{bmatrix} 10 \sin(\theta) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \tan\frac{\pi}{8} \end{bmatrix} u + \begin{bmatrix} 1 & g_d^{(p)}(x) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} d^{(w)} \\ d^{(p)} \end{bmatrix} \tag{4.16}$$

where u is the input to the model (tangent of the steering angle), $d^{(w)}$ is an external disturbance in lateral car velocity (e.g. wheel slip), and the terms $g_d^{(p)}(x)$ and $d^{(p)}$ are used to bound the error in the Taylor expansion of the dynamics.

Since our methods require the dynamics to be in polynomial form, we Taylor expand the lane position dynamics about a car angle of 0. To avoid the computational expense of high-order polynomial dynamics, we replace the term $\sin(\theta)$ in (4.16) with a low-order expansion $f^{(l)}$ and bound the approximation error with the disturbance signal $g_d^{(p)}(\cdot)d^{(p)}$. To compute this bound, we use the following sums-of-squares optimization to find the minimum bounding polynomial of degree 5 between our low-order (degree 5) dynamics and a high-order (degree 15) expansion, $f^{(h)}$:

$$\begin{aligned} \inf_{g_d^{(p)}, \sigma_1, \sigma_2} \; & \int_X g_d^{(p)} \\ \text{s.t. } & g_d^{(p)} - \left(f^{(h)} - f^{(l)}\right) - h_X \sigma_1 \in SoS \\ & g_d^{(p)} + \left(f^{(h)} - f^{(l)}\right) - h_X \sigma_2 \in SoS \\ & \sigma_1, \sigma_2 \in SoS \end{aligned} \tag{4.17}$$

With g_d included in the dynamics, we computed a viability domain and semi-autonomous controller using our method with degree 16 polynomials. The computation took approximately 1 minute. The resulting safe set is given in Fig. 4.3. For these dynamics, we can compute the exact unsafe set by simulating the system backwards in time from the corners of the failure set under maximal input and disturbance. The result is shown in grey in Fig. 4.3, where we see that our method produces a tight inner approximation to the true viability kernel. Additionally, we ran a pair of simulations (one with the semi-autonomous controller inactive and another with it active), illustrated in Fig. 4.3, of the system under a randomized input.

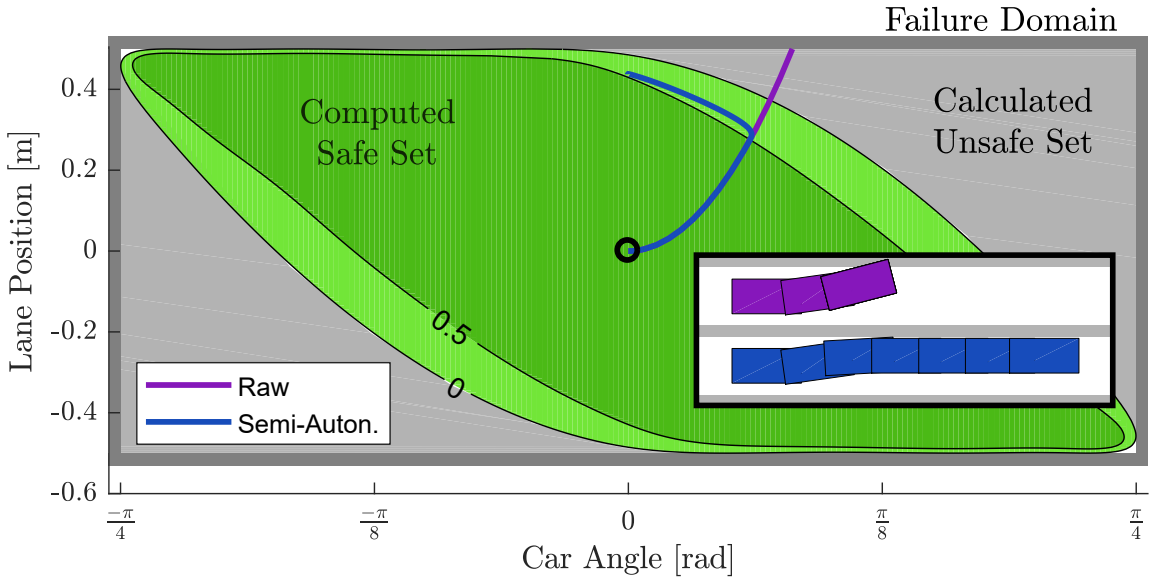

Figure 4.3: Visualization of the safe set for the Dubins car showing the 0 and 0.5 level sets of the barrier function v in light green and dark green, respectively. For this example, we can compute the exact unsafe set (complement of the viability kernel), shown here in light grey. Shown in the overlay are two simulated vehicle trajectories, one (purple) uses a randomly generated input, the other (blue) combines this input with a semi-autonomous controller (threshold value $v_m = 0.5$). We see the randomly generated controller drive off the road (fail), while the semi-autonomous vehicle remains safe. The trajectories are also plotted in state space where we see that the semi-autonomous vehicle diverges only once $v = v_m$ is reached.

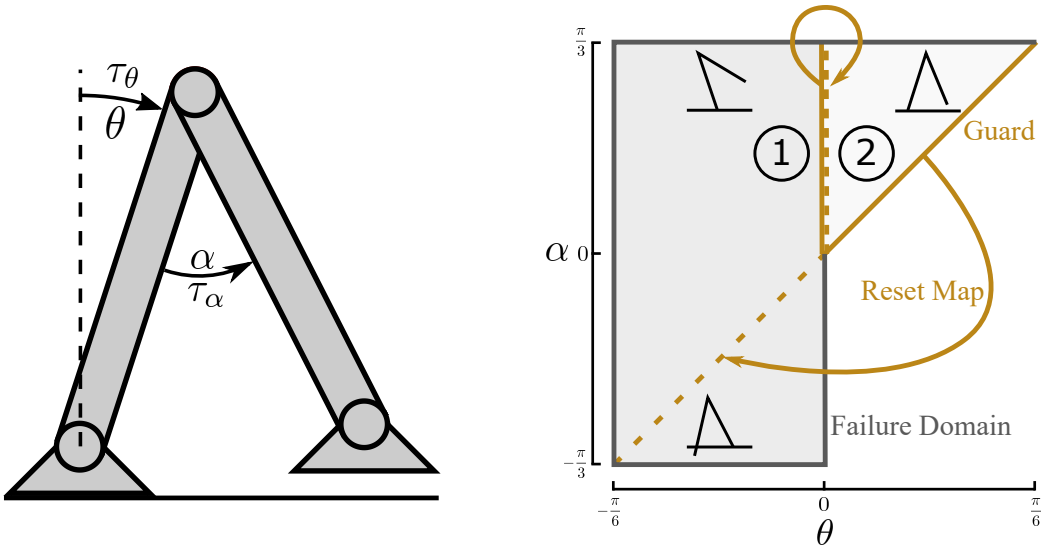

Figure 4.4: Compass gait states (right) and domains (left). The walker has two legs with mass and length parameters loosely chosen to correspond with the robot RAM*one* [89] (see supplementary code[2]). There are two bounded control inputs: a torque at the ankle and a torque between stance and swing legs. The robot has two discrete states: pre-midstance (mode 1) and post-midstance (mode 2). There is a transition defined from mode 1 to mode 2 and one defined from mode 2 to mode 1. The hybrid guard from mode 1 to mode 2 is associated with an identity reset map. The hybrid guard from mode 2 to 1 represents a touchdown event. The associated reset maps the stance leg angle to the swing leg and vice versa.

4.6.2 Compass Gait Walker

To demonstrate our method in a hybrid setting, we implement it on a compass-gait walker, shown in Fig. 4.4. The continuous and discrete dynamics of the walker are derived using implicitly constrained Newton-Euler equations, in a fashion similar to [88]. These dynamics have four continuous states representing the two leg angles and their velocities ($[\theta, \alpha, \dot{\theta}, \dot{\alpha}] = x$), and two discrete modes as shown in Fig. 4.4. There are two control inputs to this system: an ankle torque (τ_θ) and a hip torque (τ_α), bounded to be within the intervals $\tau_\theta \in [-1, 1]Nm$, $\tau_\alpha \in [-10, 10]Nm$. Due to space limitations, the derivation and expression of the dynamics is given in full detail in the supplementary code[2].

The full domain of the system (Fig. 4.4) is given by the set:

$$\{(\theta, \alpha, \dot{\theta}, \dot{\alpha}) \in [-\pi/6, \pi/6] \times [-\pi/3, \pi/3] \times [0, 5] \times [-10, 10]\} \backslash \left\{ x \mid (\alpha < 2\theta) \bigcup (\theta > 0) \right\}. \tag{4.18}$$

The domain can be understood as all states between the joint angle limits in which

[2] `https://github.com/nilssmit/IROS_2018_Safe_Control`

either the swing foot is above the ground, or the hip has not yet reached apex. To express this domain as a semi-algebraic set, we split it into into two separate modes, along the line $\theta = 0$. We add a guard in the first mode ($\theta \leq 0$) along $\theta = 0$ with an identity reset into the second mode ($\theta \geq 0$). Since $\dot{\theta} >= 0$ for all points in the state space, the state can only traverse from mode 2 to mode 1 along the touchdown guard. This allows us to remove the negativity constraint, (4.3), on the $\theta = 0$ border of mode 2.

The dynamics (f, g_u, g_d) are approximated using 5^{th} order Taylor expansions about the origin. To capture the approximation error in the dynamics, we bound the difference to a 15^{th} order expansion and add it as a disturbance in the dynamics similarly to Section 4.6.1. The reset map is approximated using a 2^{nd} order Taylor expansion about the center of its domain. The low-order approximation is necessary because the reset map takes the polynomial v as an input argument in the sums-of-squares program. Since the degree of a composition of two functions is the product of their degrees, a high-order expansion would result in a large computational penalty. The approximation error that results from this low dimensional reset map cannot be accounted for in our current formulation, but is something that we plan to explore in future work. The Taylor expanded dynamics, resets and guards are given in the supplementary code[2].

Under these dynamics, our method takes approximately 2h to find a viability domain and semi-autonomous controller using degree 4 polynomials. The resulting safe set is visualized in Fig. 4.5. Additionally, a MATLAB application for simulating the system under manual control input can be found in the supplementary code[2].

4.7 Conclusion

This chapter presents a method to construct guaranteed safe semi-autonomous controllers for hybrid systems. The resulting controller guarantees viability and allows for arbitrary input when viability is not at risk. It can be computed directly, and does not require real-time solution of optimization problems. The method is evaluated on a compass gait walking model with 4 states and 2 hybrid modes. Safe interaction with the physical world is a primary goal of robust control. Future work is aimed at hardware evaluation. Remaining challenges to this end include increasing the model complexity, and bounding dynamic uncertainty observed in physical data.

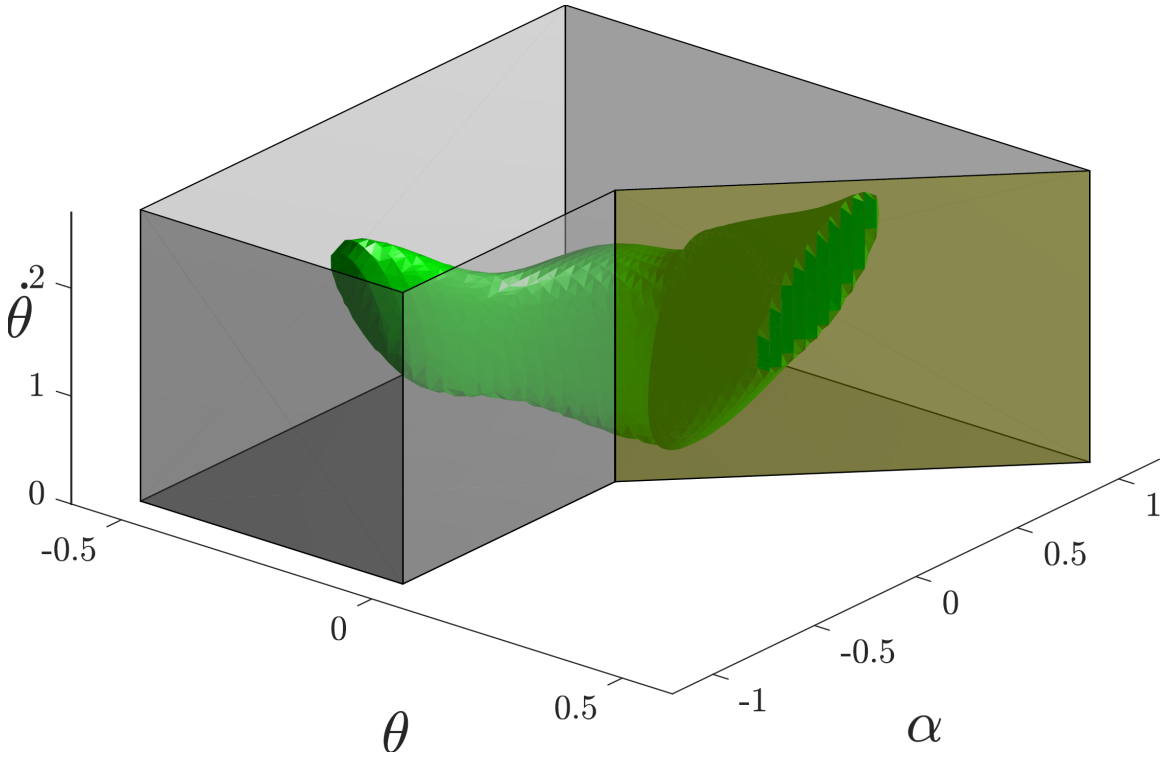

Figure 4.5: Safe set (shown in green) for the compass gait walker visualized as a 3D slice of the 4D state space with a stationary swing leg ($\dot{\alpha} = 0$). The yellow plane represents the touchdown guard, and the gray planes represent the edges of the failure set, i.e. the state must stay within these boundaries to remain viable. The viable set does not intersect the failure set, but does intersect the guard. Note that along the guard, increasing $\dot{\theta}$ requires an increase in α for the state to be viable. This matches the intuition that larger step lengths should be used for higher walking speeds [75]. Also note that the minimum stance leg speed is large for large stance leg angles. For lower speeds, the stance leg torque is insufficient to get the walker over mid-stance.

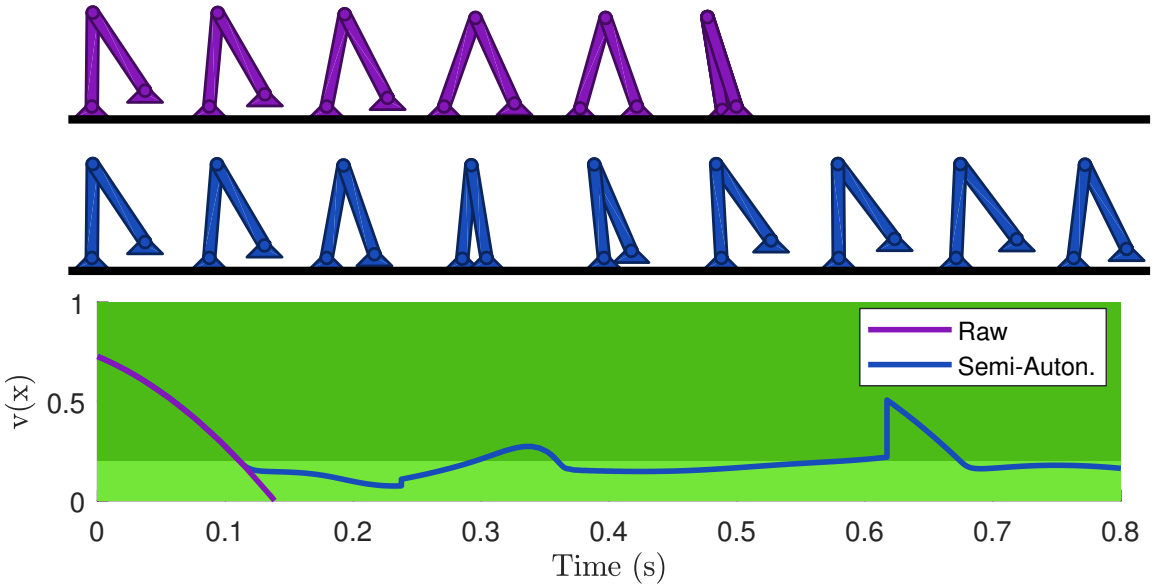

Figure 4.6: Two simulated trajectories for the compass gait model simulated using the non-Taylor-expanded dynamics. The first, in purple, uses a random control input, the other, in blue, uses our semi-autonomous controller in combination with this input (threshold value $v_m = 0.2$). In the trajectory stills we see that the raw trajectory fails by falling backwards ($\dot{\theta} < 0$), while the semi-autonomous trajectory keeps walking. Also shown is the barrier function value ($v(x)$) over time. Note the semi-autonomous controller deviates from nominal only once the threshold value (shown in light green) is reached.

CHAPTER V

Walking with Confidence: Safety Regulation for Full Order Biped Models

This chapter aims to address the scaling limitations faced by the safety regulator of the preceding section. By using a dimensionality-reduction approach based on hybrid zero dynamics, this chapter scales the safety regulator to a 10-dimensional walking model. The contents of this chapter have been accepted for publication in Robotics and Automation Letters 2019 [90], and are presented here as originally published.

5.1 Introduction

Avoiding falls is a safety critical and challenging task for legged robotic systems. This challenge is compounded by strong limits on the available actuation torques; particularly at the ankle or ground contact point. These limits in actuation mean that the motion of a legged robot is often dominated by its mechanical dynamics, which are hybrid, nonlinear, and unstable. A consequence of these limitations is that a controller might be required to take a safety preserving action well before the moment a failure occurs.

Consider, for example, a bipedal robot that just entered single stance during a fast walking gait. The robot is pivoting dynamically over the stance foot and can only apply limited ankle torques to control its motion. To catch the robot again, the swing foot needs to be brought forward rapidly and be placed well in front of the robot. If the forward velocity of the robot and hence the pivoting motion is too fast, there will not be enough time to complete this foot placement far enough in front of the stance leg to slow the robot down [73]. As a result, the robot's speed increases further, leaving even less time for leg swing in the subsequent steps. The robot might manage to complete another couple of strides, but at this point a fall is inevitable

and no control action can prevent it.

Knowing the limits of safe operation is akin to knowing the set of states from which falls, even in the distant future, can be avoided. Such knowledge is valuable for many reasons. Knowing that a fall is inevitable is useful in itself, as it allows a robot to brace for the imminent impact. Knowing the distance from the border of the safe set could allow a robot to estimate the set of impulses that can be withstood without failing. This would allow it to judge whether or not it can safely interact with the environment in a given situation; for example, to push a cart while walking. Most importantly, this knowledge is valuable due to the flexibility it can create. Rather than stabilizing the robot motion along a specified trajectory, one could imagine controllers that are adaptive to adjust to the environment, to maximize performance, or to fulfill a secondary task such as pointing a sensor onto a target. Any of these secondary tasks can be pursued as long as the state of the robot is within the safe set.

In this context, a representation of the set of safe states enables the construction of a regulator that monitors the system state and takes safety preserving actions only when the robot is at risk of failure [106]. Such a regulator could guarantee safe operation, while allowing a secondary control system to behave flexibly as long as safety is not threatened.

Identifying such safety limits, however, is a challenging problem for nonlinear and hybrid systems. A promising tool for identifying the safety limits of a legged robotic system is sums-of-squares (SoS) optimization [70]. This approach uses semi-definite programming to identify the limits of safety in the state space of a system as well as associated controllers for a broad class of nonlinear [56, 37, 50] and hybrid systems [72, 87]. These safe sets can take the form of *reachable sets* (sets that can reach a known safe state) [49, 87, 56] or *invariant sets* (sets whose members can be controlled to remain in the set indefinitely) in state space [105, 72, 71]. However, the representation of each of these sets in state space severely restricts the size of the problem that can be tackled by these approaches. To accommodate this limitation, sums-of-squares analysis has been primarily applied to reduced models of walking robots: ranging from spring mass models [111], to inverted pendulum models [49, 96] and to inverted pendulum models with an offset torso mass [71]. The substantial differences between these simple models and real robots causes difficulty when applying these results to hardware.

A contrasting approach to designing stable controllers for high dimensional, underactuated robot models uses hybrid zero dynamics (HZD) [104]. In this approach,

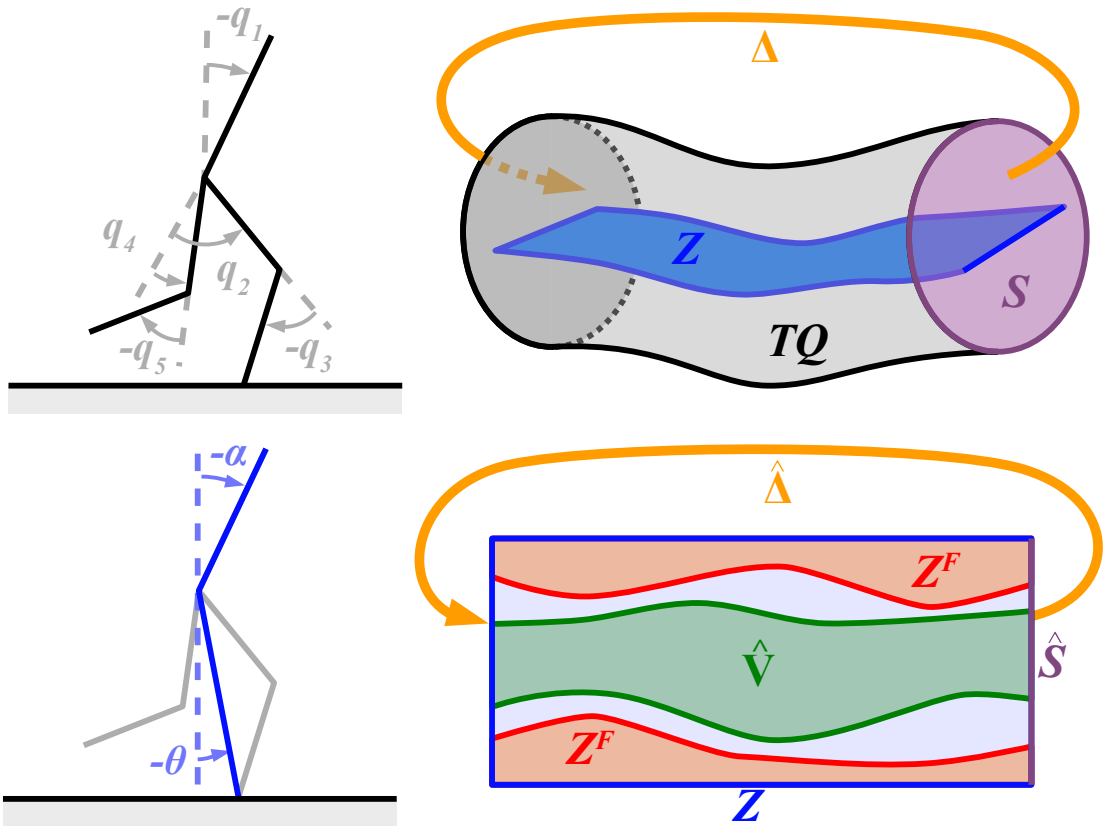

Figure 5.1: Generating safety guarantees for a high dimensional robot (illustrated on Rabbit [18]). The state-space of the full robot is given in the top right figure, where TQ is the tangent space on Q, S is the hybrid guard representing foot touchdown, and Δ is the corresponding discrete reset map. Using feedback linearization, we restrict our states to lie on a low-dimensional manifold Z, reducing the state-space dimension to an amenable size for sums-of-squares analysis. This manifold is parameterized by the underactuated degrees of freedom of the robot θ, as well as a set of shaping parameters α. The shaping parameters can be modified in real-time by a control input, allowing for a broad range of behaviours on Z. To guarantee safety on Z we find the set of unsafe states Z^F from which the state may leave the manifold (for instance due to motor torque limits). We then use sums-of-squares tools [70] to find a control invariant set $\hat{V} \subset Z \setminus Z^F$. This control invariant set can be used to define a semi-autonomous, guaranteed safe controller for the full robot dynamics.

feedback linearization is used to drive the actuated degrees of freedom of the robot towards a lower dimensional hybrid zero dynamics manifold. This manifold is specified as the zero levelset of a configuration-dependent output vector and represents the motion of the robot in its underactuated degrees of freedom.

Significant progress has been made in the generation of safety certificates for HZD controllers. Much of this work [4, 41, 66, 67, 68, 65] relies on the Poincaré stability of a periodic limit cycle in order to generate safety guarantees. This reliance is restrictive, as it precludes behaviors that would leave the neighborhood of the limit cycle. Recent work has been done to extend the range of safe HZD behaviours beyond a single limit cycle neighborhood [63, 101, 102, 5]. In [63, 101, 102], the controller is allowed to discretely switch between a family of periodic gaits. Safety is then ensured using a dwell time constraint that limits how frequently switching can occur. In [5], a combination of HZD and finite state abstraction is used to safely regulate forward speed of a fully-actuated bipedal robot. This method requires all robot degrees of freedom to be actuated in order to construct safety certificates. Our approach shares strong similarities with each of these works, but allows for continuous variation within the family of behaviours, and applies to underactuated robotic systems.

In this chapter we build on these two broad approaches to safety and control synthesis for legged robotic systems. To combine the full-model accuracy of hybrid zero dynamics and the set-based safety guarantees of sums-of-squares programming, we propose the following approach (Fig. 5.1). First, we use hybrid zero dynamics to map the full order dynamics to a low dimensional hybrid manifold. We control the dynamics on the manifold using a set of *shaping parameters*, which are modified in continuous time to modify robot behaviour. We then use sums-of-squares programming to find a subset of this manifold which can be rendered forward control invariant. Once this subset is found on the low dimensional manifold, a regulator can be constructed that allows for free control of the manifold dynamics when safety is not at risk, but switches to a safety preserving controller when safety is threatened.

The approach is presented in a general form that extends to a large class of underactuated bipedal robots. Throughout the chapter, an example implementation is given for a 10-dimensional model of the robot Rabbit [18] and a tracking task is used to illustrate semi-autonomous safe control. To the best of our knowledge, this is the highest dimensional walking robot model for which set-based safety guarantees have been generated thus far.

The rest of this chapter is organized as follows: Section 5.2 formally defines the assumptions and objective of this chapter. The next two sections describe our method.

Section 5.3 constructs a low dimensional zero dynamics manifold with control input. In Section 5.4 we present a sums-of-squares optimization which finds a control invariant subset of the manifold that avoids a designated set of unsafe states. Section 5.5 describes the results of our implementation on the robot Rabbit [18], and conclusions are presented in Section 5.6.

5.2 Problem Setup

5.2.1 Robot Model

For simplicity, we apply similar modeling assumptions to those made in [104]. That is, the robot is modeled as a planar chain of rigid links with mass. Each joint is directly torque actuated except for the point of contact with the ground, leading to one degree of underactuation for a planar model. The full configuration of the robot is given by the set of joint angles $q = \{q_1, \ldots, q_{n_q}\} \in Q \subset \mathbb{R}^{n_q}$. We next define the set of *feasible* configurations $\tilde{Q} \subset Q$ (similarly to [105]):

Definition 1. A configuration is *feasible* if the joint angles satisfy actuator limits, and only foot points are touching the ground (i.e. the robot has not fallen over).

Using the method of Lagrange, we can obtain a continuous dynamic model of the robot during swing phase:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t). \tag{5.1}$$

where $x(t) = [q^\top(t), \dot{q}^\top(t)]^\top \in TQ \subset R^{2n_q}$ denotes the tangent space of Q, $u(t) \in U$, U describes the permitted inputs to the system, and t denotes time.

We assume that an instantaneous and impulsive impact occurs each time the swing foot hits the ground, with the stance leg leaving the ground immediately after impact. As in [104], we can construct a reset map for the state after impact:

$$x(t^+) := \Delta(x(t^-)) \tag{5.2}$$

$$= \begin{bmatrix} \Delta_q q(t^-) \\ \Delta_{\dot{q}}(q(t^-))\dot{q}(t^-) \end{bmatrix}. \tag{5.3}$$

Here the superscript plus indicates the time just after the event and the superscript minus indicates the time just before the event. $\Delta : TQ \to TQ$ is the reset map of the robot state. $\Delta_q \in \mathbb{R}^{n_q \times n_q}$ is a coordinate transformation matrix that swaps the swing leg and the stance leg after impact. $\Delta_{\dot{q}} : Q \to \mathbb{R}^{n_q \times n_q}$, is the configuration-dependent

reset map of the configuration velocities.

This equation holds true for all states in $S \subset TQ$, which is called the *guard* of the hybrid system, and represents the states of the robot with zero swing foot height and downwards swing foot velocity. Any time the state of the robot enters S, the reset event must occur.

Example 1. The configuration q for Rabbit is shown in Figure 5.1 (top left). $\tilde{Q}$ is the set of robot configurations in which only foot points intersect the ground and all joints are within the limits: $q_1, q_2, q_4 \in [-\pi/2, \pi/2]$, $q_3, q_5 \in [-\pi/2, 0]$. When the swing foot intersects with the ground, we enter the guard S. This causes an impulse to be transmitted to the colliding foot, and the swing and stance feet swap. The impulse and coordinate swap are given by Δ. The joint torques torques are saturated to take values in the interval $U = [-30\,\mathrm{Nm}, 30\,\mathrm{Nm}]^4$. All kinematic and inertial properties of the model are given in [18].

5.2.2 Safety

In this chapter, *safety* is defined as keeping the configuration *feasible* for all time (i.e. $q(t) \in \tilde{Q}, \forall t$). To guarantee safety, this chapter finds a *viability domain* [105]:

Definition 2. A *viability domain* $V \subset \mathbb{R}^{2n_q}$ is any set satisfying $V \subset T\tilde{Q}$ which is also forward control invariant. That is, there exists a Lipschitz state feedback controller $u_s : T\tilde{Q} \to U$, such that for every initial condition $x_0 \in V$, the execution of the system from the initial condition remains in V for all time $t \in [0, \infty)$. We refer to any feedback controller that is able to ensure that the system is forward control invariant as an *Autonomous Viable Controller*.

The forward control invariance property ensures that any state that begins within a viability domain V can be controlled to remain within the domain. Since V contains only feasible configurations ($V \subset T\tilde{Q}$), we know that safety can be maintained by at least one controller from all states in V.

Once a viability domain is found, we use it to construct a semi-autonomous, safety preserving controller. Given an initial state within V, a user defined control input is applied without modification to the system. The state of the system is then continuously monitored. If the state approaches the boundary of the viability domain, the control input is overridden by an autonomous viable controller. This gives the user full control over the system until safety is threatened, at which point, safety is automatically enforced. Once safety is no longer at risk, control is returned to the user.

5.2.3 Goal

Using these definitions, we state our objective as:

1. Find a viability domain and a corresponding autonomous viable controller.
2. Use this domain and autonomous viable controller to construct a semi-autonomous viable controller.

5.3 Controlled Hybrid Zero Dynamics Manifold

We intend to use sums-of-squares optimization to achieve these objectives. However, the state-space dimension of realistic robot models far exceeds the limits of this tool. For instance, the state-space of the benchmark model Rabbit [18] has dimension 10, while many sums-of-squares problems become computationally challenging above dimension 6 [71]. In this section, we show how the the state-space dimension can be reduced to a feasible size using the idea of hybrid zero dynamics [104].

5.3.1 Shaping Parameters

The hybrid zero dynamics approach uses feedback linearization to drive the actuated degrees of freedom onto a low-dimensional manifold specified by a set of user-chosen outputs, which depend on the robot configuration $q \in Q$. We modify this approach by making these outputs also depend on a set of time varying shaping parameters $\alpha(t) \in A \subset \mathbb{R}^{n_\alpha}$. The shaping parameters α are used in this chapter to provide an input within the manifold dynamics. By varying α continuously over time, the user can change the hybrid zero dynamics manifold to modify the robot behaviour in real-time. The idea of modifying the HZD manifold in real-time is similar to prior work [92, 63], where parameters are allowed to change discretely once per robot step. In contrast, α can vary throughout the step, enabling a rapid response to external input without waiting for the next discrete update.

We define the dynamics of α as:

$$\dot{x}_\alpha(t) = f_\alpha(x_\alpha(t)) + g_\alpha(x_\alpha(t))u_\alpha(t), \tag{5.4}$$

where $x_\alpha(t) = [\alpha^\top(t), \dot{\alpha}^\top(t)]^\top \in TA$, $u_\alpha(t) \in U_\alpha \subset \mathbb{R}^{n_\alpha}$ are the shaping parameter inputs (with permitted values U_α), and t denotes time. We require that α has vector relative degree two under these dynamics. We assume a trivial discrete update for the shaping parameters when the robot state hits a guard: $x_\alpha(t^+) = x_\alpha(t^-)$.

Example 2. As shown in the bottom left of Figure 5.1, we use a single shaping parameter $\alpha(t) \in [-\pi/2, \pi/2]$ to modify the desired pitch angle of Rabbit. Note that this choice is somewhat arbitrary; α could instead modify properties such as step length or center of mass height. We define the dynamics of α as follows:

$$\frac{d}{dt}\begin{bmatrix}\alpha(t)\\ \dot{\alpha}(t)\end{bmatrix} = \begin{bmatrix}\dot{\alpha}(t)\\ 0\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix} u_\alpha(t), \tag{5.5}$$

where u_α represents the user-controlled pitch acceleration.

5.3.2 Constructing the Manifold

In this subsection, we incorporate these shaping parameters in the construction of the hybrid zero dynamics manifold described in [104]. Throughout the section, we use $\mathcal{L}_f^x$ and $\mathcal{L}_g^x$ to represent the Lie derivatives in TQ with respect to f and g, and $\mathcal{L}_{f_\alpha}^{x_\alpha}$ and $\mathcal{L}_{g_\alpha}^{x_\alpha}$ to represent the Lie derivatives in TA with respect to f_α and g_α (where we drop the arguments).

We begin by using a set of outputs: $h : Q \times A \to \mathbb{R}^{n_u}$ to implicitly define the hybrid zero dynamics manifold as:

$$\begin{aligned} Z := \{(q, \dot{q}, \alpha, \dot{\alpha}) \in TQ \times TA \,|\, h(q, \alpha) = 0, \\ (\mathcal{L}_f^x h)(q, \alpha, \dot{q}) + (\mathcal{L}_{f_\alpha}^{x_\alpha} h)(q, \alpha, \dot{\alpha}) = 0\} \end{aligned} \tag{5.6}$$

These outputs must satisfy hypotheses similar to HH 1-4 in [104], and the resulting manifold Z must satisfy the *hybrid invariance condition*:

$$\begin{bmatrix}\Delta(x(t^-))\\ x_\alpha(t^-)\end{bmatrix} \in Z \quad \forall \begin{bmatrix}x(t^-)\\ x_\alpha(t^-)\end{bmatrix} \in Z \cap (S \times A)\,. \tag{5.7}$$

Provided these conditions are met, we can use the results in [85, Chapter 9.3.2] to show that Z is a smooth submanifold in $TQ \times TA$ of dimension $n_z = 2(n_q - n_u + n_\alpha)$. In addition, the control input $u^* : TQ \times TA \times U_\alpha \to U$ given by:

$$\begin{aligned} u^*(x, x_\alpha, u_\alpha) = -(\mathcal{L}_g^x(\mathcal{L}_f^x h + \mathcal{L}_{f_\alpha}^{x_\alpha} h))^{-1}\Big(\mathcal{L}_f^x(\mathcal{L}_f^x h + \mathcal{L}_{f_\alpha}^{x_\alpha} h) + \\ + \mathcal{L}_{g_\alpha}^{x_\alpha}(\mathcal{L}_f^x h + \mathcal{L}_{f_\alpha}^{x_\alpha} h)u_\alpha + \mathcal{L}_{f_\alpha}^{x_\alpha}(\mathcal{L}_f^x h + \mathcal{L}_{f_\alpha}^{x_\alpha} h)\Big) \end{aligned} \tag{5.8}$$

renders Z invariant under the hybrid dynamics of the robot (note the right hand side arguments are suppressed to simplify presentation).

As in hypothesis HH 3 in [104], we define a set of phasing coordinates $\theta : Q \to \mathbb{R}^{n_q - n_u}$ which represent the underactuated degrees of freedom of the robot. Using these coordinates, we can parameterize the on-manifold state of the robot $\hat{x}(t) \in Z$ as: $\hat{x}(t) = [\theta(q)^\top, \dot{\theta}(q,\dot{q})^\top, \alpha^\top, \dot{\alpha}^\top]^\top$ (where we have suppressed the time dependence on the right hand side). The continuous dynamics under this parameterization are then:

$$\dot{\hat{x}} = \begin{bmatrix} \dot{\theta} \\ \mathcal{L}_f^x \mathcal{L}_f^x \theta + \mathcal{L}_g^x \mathcal{L}_f^x \theta u^* \\ f_\alpha + g_\alpha u_\alpha \end{bmatrix} = \hat{f}(\hat{x}) + \hat{g}(\hat{x}) u_\alpha, \tag{5.9}$$

where we have suppressed the time dependence. The discrete manifold dynamics are given by:

$$\hat{x}(t^+) = \hat{\Delta}(\hat{x}(t^-)), \qquad \forall \hat{x}(t^-) \in \hat{S}, \tag{5.10}$$

where t^- is the state before impact, and the manifold guard and reset ($\hat{S}$ and $\hat{\Delta}$) are defined as:

$$\hat{S} = Z \cap (S \times A) \tag{5.11}$$

$$\hat{\Delta}(\hat{x}(t^-)) = \begin{bmatrix} \theta(\Delta_q(q(t^-))) \\ \frac{\partial \theta}{\partial q}(\Delta_q(q(t^-)))\Delta_{\dot{q}}(q(t^-))\dot{q}(t^-) \\ x_\alpha(t^-) \end{bmatrix}. \tag{5.12}$$

Example 3. We begin by using the trajectory optimization toolbox FROST [38] to find a time-varying, periodic walking trajectory: $q^{Fr} : [0, t_{max}] \to Q$. For this trajectory, the stance leg angle of the robot: $\theta(q) = -q_1 - q_2 - \frac{q_3}{2}$ is monotonic in time and varies from θ_{min} to θ_{max}. This allows us to define a phasing function $t_\theta : [\theta_{min}, \theta_{max}] \to [0, t_{max}]$ which satisfies $q^{Fr}(t_\theta(\theta(q^{Fr}(t)))) = q^{Fr}(t)$ (i.e. t_θ maps from points in the state space to points along the trajectory).

We modify the pitch angle of the FROST trajectory using the shaping parameter α, giving us the output function:

$$h(q,\alpha) = \begin{bmatrix} q_1 - q_1^{Fr}(t_\theta(\theta(q))) - \alpha \\ q_3 - q_3^{Fr}(t_\theta(\theta(q))) \\ q_4 - q_4^{Fr}(t_\theta(\theta(q))) + \alpha \\ q_5 - q_5^{Fr}(t_\theta(\theta(q))) \end{bmatrix} + h_m(\theta(q), \alpha). \tag{5.13}$$

Here we also added the function $h_m : Q \times A \to \mathbb{R}^4$ which is chosen to ensure satis-

faction of the *hybrid invariance condition* (5.7). This technique for ensuring hybrid invariance is similar to the procedure given in [92]. See Appendix A for a more detailed derivation of this condition.

The guard of our HZD manifold Z is given as $\hat{S} = \{\hat{x} \mid \theta = \theta_{max}, \dot{\theta} > 0\}$ and the reset is defined as in (5.12).

5.3.3 Safety on the Manifold

We now revisit the safety criteria from Section 5.2.2 under the assumption that our state is controlled to lie on Z. For the biped to be safe, we require that the manifold state remains in the feasible set $\tilde{Q}$, and that the state does not leave the manifold (either by leaving the manifold boundary, or by encountering actuator limits when trying to stay on Z). We define the *unsafe states* $Z^F \subset Z$ as the union of:

- The *infeasible* states: $((TQ \setminus T\tilde{Q}) \times TA) \cap Z$
- The states that *leave the manifold boundary*, i.e. all members of the boundary set ($\partial Z = \{\hat{x} \in Z \mid q_0(\hat{x}) \in \partial Q \text{ or } \alpha \in \partial A\}$) which do not lie on a guard, and that have an outward velocity.
- The states *requiring unattainable actuation* to remain on Z, i.e. all states $(x, x_\alpha) \in Z$ for which $u^*(x, x_\alpha, u_\alpha) \notin U,\ \forall u_\alpha \in U_\alpha$.

Additionally we define the state-dependent set of realizable shaping parameter inputs $\hat{U} : TQ \times TA \to 2^{U_\alpha}$, as $\hat{U}(x, x_\alpha) = \{u_\alpha \in U_\alpha \,|\, u^*(x, x_\alpha, u_\alpha) \in U\}$ (where 2^{U_α} denotes the set of all subsets of U_α).

Provided we constrain the manifold state to avoid Z^F, and constrain the shaping parameter input to lie within $\hat{U}$, our safety criteria is maintained.

Our goal from Section 5.2.3 can now be re-stated as:

1. Find a viability domain on Z that does not intersect Z^F, and an autonomous viable controller $\hat{u}_s : Z \to \hat{U}$.
2. Use this domain and autonomous viable controller to construct a semi-autonomous controller.

Example 4. For the Rabbit example, the set of states that leave the manifold boundary are given by $Z_{LMB} = \{\hat{x} \mid \alpha = \pi/2,\ \dot{\alpha} > 0\} \cup \{\hat{x} \mid \alpha = -\pi/2,\ \dot{\alpha} < 0\}$. All other states on the manifold boundary either lie on a guard ($\theta = \theta_{max}, \dot{\theta} > 0$), or flow inwards.

We use sampling and fitting to find a region $Z_{Lim} \subset Z$ where the actuator torque limits can be satisfied for some u_α. We then define our unsafe set (see Fig. 5.2):

$$Z^F = (((TQ \setminus T\tilde{Q}) \times TA) \cap Z) \cup Z_{LMB} \cup (Z \setminus Z_{Lim}). \tag{5.14}$$

The set of attainable inputs $\hat{U}$ is given by the minimum and maximum values of u_α at each sample point $(x, x_\alpha) \in Z$ that satisfy $u^*(x, x_\alpha, u_\alpha) \in U$.

5.4 Hybrid Control Invariant Set

This section outlines how the low dimensional safety problem from Section 5.3.3 can be solved using sums-of-squares optimization [70]. Broadly, the sums -of-squares approach enforces constraints of the form $p \geq 0$ (where p is a function) by constraining p to be a sum-of-squares polynomial, i.e. $p = \sum_i p_i^2$ (where p_i are polynomials). We refer to this constraint as $p \in SoS$.

We begin by showing how the sets and dynamics from the preceding section can be represented using polynomials. We next define a bilinear semi-definite program for finding a viability domain, and describe the alternation used to solve it. Finally, we construct a guaranteed safe semi-autonomous controller for the full robot, based on this viability domain.

5.4.1 Polynomial Representation

For the dynamics of the system to be used inside our sums-of-squares program, they must be represented in a polynomial form. In particular, we require polynomial representations of the functions $\hat{f}, \hat{g}, \hat{\Delta}$ and the sets $\hat{S}, Z^F, \hat{U}$. Since these sets and functions can contain trigonometric as well as rational terms in their definition, we rely on approximate representations. It is important to take care to ensure that the safety guarantee is preserved under approximation.

To generate polynomial approximations and verify bounding relations, we use sampling to obtain the exact function values over a dense grid in the state space. This sampling approach is made tractable by the reduction in dimension of the previous section. In our example, this reduces the dimension that must be sampled from 10 to 4. We use a $30 \times 30 \times 30 \times 30$ sample grid to fit and bound the polynomials. The bounds are then verified using a dense set of randomly generated test points.

We begin by sampling $\hat{f} : Z \to \mathbb{R}^{n_z}$ and $\hat{g} : Z \to \mathbb{R}^{n_z \times n_\alpha}$ over our grid of points in Z. Least-squares fitting can then be used to obtain the corresponding polynomial

representations: $\hat{f}_p$ and $\hat{g}_p$. To account for the approximation error in the continuous dynamics functions, we introduce a set of error-bounding polynomials $\hat{e}_p : Z \to \mathbb{R}^{n_z}$ which satisfy:

$$\hat{e}_p(\hat{x}) \geq \left| \hat{f}(\hat{x}) - \hat{f}_p(\hat{x}) + (\hat{g}(\hat{x}) - \hat{g}_p(\hat{x}))\, \hat{u} \right|, \tag{5.15}$$

for all $\hat{x} \in Z$ and $\hat{u} \in \hat{U}$ where the inequality and absolute value are taken element-wise. These polynomials can be found using a linear program that minimizes the integral of $\hat{e}_p$ subject to (5.15) enforced at our set of sample points.

To represent sets in polynomial form, we require them to take the form of semi-algebraic sets (i.e. a set $X \subset Y$ is defined as $X = \{y \in Y \mid h_i(y) \geq 0,\ \forall i = 1, \dots n\}$, where $h : Y \to \mathbb{R}^n$ is a collection of polynomials). We use a bounding set to approximate the reset map $\hat{\Delta} : \hat{S} \to Z$ in a conservative manner. That is, we find a set $R_p \subset Z \times Z$ that bounds all possible reset behaviours:

$$(\hat{x}, \hat{\Delta}(\hat{x})) \in R_p,\ \forall \hat{x} \in \hat{S}. \tag{5.16}$$

The sets $\hat{S}$ and Z^F are represented with semi-algebraic outer approximations as follows: $Z_p^F \supset Z^F$, $\hat{S}_p \supset \hat{S}$. We define the sets $R_p, Z_p^F, \hat{S}_p$ using the respective polynomials: $h_R : Z \times Z \to \mathbb{R}^{n_{hr}}$, $h_F : Z \to \mathbb{R}^{n_{hf}}$, $h_S : Z \to \mathbb{R}^{n_{hs}}$. The space of feasible inputs $\hat{U}$ can be approximated using a state-dependent box constraint:

$$\hat{u}_{min}(\hat{x}) \leq \hat{u}(\hat{x}) \leq \hat{u}_{max}(\hat{x}),\ \forall \hat{x} \in Z \setminus Z_p^F \tag{5.17}$$

where $\hat{u}_{min}, \hat{u}_{max} : Z \setminus Z_p^F \to U_\alpha$ are polynomial input bounds, and the inequality is taken element-wise. The set of inputs that satisfy this box constraint is denoted by $\hat{U}_p$.

5.4.2 Optimization Formulation

We use an optimization similar to that of the previous chapter to find the largest possible viability domain $\hat{V} \subset Z \setminus Z^F$ for our hybrid zero dynamics system. We represent $\hat{V}$ as the zero super-levelset of a polynomial function $\hat{v} : Z \to \mathbb{R}$ (i.e. $\hat{V} = \{\hat{x} \in Z \mid \hat{v}(\hat{x}) \geq 0\}$), and represent the autonomous viable controller using a polynomial function $\hat{u}_s : Z \to \mathbb{R}^{n_\alpha}$. To enforce the viability of $\hat{V}$ according to Definition 2, we require $\hat{v}$ and $\hat{u}_s$ to satisfy four conditions:

Viability Conditions.

1. $\hat{V}$ does not intersect Z^F (i.e. $\hat{v}(\hat{x}) < 0,\ \forall \hat{x} \in Z^F$)

2. All states that are contained in both the guard and $\hat{V}$ must be mapped to a state in $\hat{V}$ (i.e. $\hat{v}(\hat{\Delta}(\hat{x})) \geq 0,\ \forall \hat{x} \in \{\hat{x} \in \hat{S} \mid \hat{v}(\hat{x}) \geq 0\}$)

3. At the boundary of $\hat{V}$ (i.e. where $\hat{v}(\hat{x}) = 0$), the state flows inward under the controller $\hat{u}_s$ (i.e. $\frac{d\hat{v}}{dt} > 0$)

4. The autonomous safe controller must satisfy the input bounds within the safe set (i.e. $\hat{u}_s(\hat{x}) \in \hat{U},\ \forall \hat{x} \in \hat{V}$)

Condition 1 ensures that states can not leave the viability domain by simply leaving the space Z. Condition 2 ensures that states can not leave the viability domain when traversing a guard. Condition 3 ensures that states cannot leave the viability domain under the continuous dynamics of the system. Finally, Condition 4 ensures that our controller respects the robot torque constraints. Each of these conditions are ensured with a corresponding sums-of-squares constraint, giving us:

SoS Constraint 1. (Viability Condition 1)

$$-\hat{v} - \sigma_1 h_F \in SoS$$

Here $\sigma_1 : Z \to \mathbb{R}^{1 \times n_{hf}} \in SoS$ are sums-of-squares polynomials that relax the positivity constraint outside Z_p^F. We refer to such polynomials as *s-functions*.

SoS Constraint 2. (Viability Condition 2)

$$\hat{v}^+ - \hat{v}^- - \sigma_2 h_R - \sigma_3 h_S^- \in SoS$$

Here $\sigma_2 : Z \times Z \to \mathbb{R}^{1 \times n_{hr}}, \sigma_3 : Z \to \mathbb{R}^{1 \times n_{hs}} \in SoS$ are s-functions. The superscripts $-$ and $+$ indicate whether a function is evaluated using the first $(-)$ or second $(+)$ argument of $h_R : Z \times Z \to \mathbb{R}^{n_{hr}}$. That is, this constraint enforces: $\hat{v}(\hat{x}^+) - \hat{v}(\hat{x}^-) - h_R(\hat{x}^-, \hat{x}^+)\sigma_2(\hat{x}^-, \hat{x}^+) - h_S(\hat{x}^-)\sigma_3(\hat{x}^-) > 0,\ \forall (\hat{x}^-, \hat{x}^+) \in Z \times Z$. Note that the addition of the σ_3 term is not strictly necessary, since points in $\hat{S}_p$ must lie in R_p. However, this term can help relax the constraint when points in R_p lie outside $\hat{S}_p$.

SoS Constraint 3. (Viability Condition 3)

$$\mathcal{L}_{\hat{f}_p}^{\hat{x}} \hat{v} + \mathcal{L}_{\hat{g}_p}^{\hat{x}} \hat{v} \hat{u}_s + \sum_{j=1}^{n_z} q_j + \hat{v}\lambda + \sigma_4 h_F \in SoS$$

$$q - \mathcal{L}_{\hat{e}_p}^{\hat{x}} \hat{v} + \sigma_5 h_F \in SoS$$

$$q + \mathcal{L}_{\hat{e}_p}^{\hat{x}} \hat{v} + \sigma_6 h_F \in SoS$$

Here $\sigma_4 : Z \to \mathbb{R}^{1 \times n_{hf}} \in SoS$ and $\sigma_5, \sigma_6 : Z \to \mathbb{R}^{n_d \times n_{hf}}$ are s-functions that relax the constraint inside Z_p^F, and $\lambda : Z \to \mathbb{R}$ is a slack polynomial that can relax the constraint whenever $\hat{v} \neq 0$. The polynomials $q : Z \to \mathbb{R}^{n_z}$ are used to bound the effects of the dynamics error $\hat{e}_p$ on the time derivative of $\hat{v}$.

SoS Constraint 4. (Viability Condition 4)

$$\hat{u}_s - \hat{u}_{min} + h_F \sigma_7 \in SoS$$
$$-\hat{u}_s + \hat{u}_{max} + h_F \sigma_8 \in SoS$$

Here $\sigma_7, \sigma_8 : Z \to \mathbb{R}^{n_\alpha} \in SoS$ are s-functions that relax the constraint inside Z_p^F.

The desired objective of our optimization is to maximize the volume of $\hat{V}$. This volume is difficult to compute exactly for an arbitrary $\hat{v}$, since the domain of integration is given by a semi-algebraic set. We propose an analytically tractable approximation to this objective:

$$\int_Z \hat{v}(\hat{x}) d\hat{x}. \tag{5.18}$$

This objective is combined with the following constraint in order to approximate the volume of $\hat{V}$:

SoS Constraint 5. (Objective Constraint)

$$1 - \hat{v} \in SoS.$$

To understand how this objective and constraint approximate the volume of $\hat{V}$, take a continuous function $\hat{v}$ that satisfies the constraints of the previous section. For every point $\hat{x}$ not in the set Z^F, the value $\hat{v}(\hat{x})$ is constrained only by Constraint (5). This means that $\hat{v}(\hat{x})$ can increase to a value of 1 for points inside $\hat{V}$, and $\hat{v}(\hat{x})$ increases to a value of 0 for points outside this set. As a result, $\hat{v}$ approaches the indicator function over $\hat{V}$, and the integral in the objective function approaches the volume of $\hat{V}$.

Combining the constraints and objective, we arrive at the following sums-of-squares problem:

$$\sup_{\substack{\hat{v},\hat{u}_s,q,\lambda \\ \sigma_1,\ldots,\sigma_6}} \int_Z \hat{v}(\hat{x})d\hat{x} \tag{5.19}$$
$$\text{s.t. SoS Constraints } 1-5,$$
$$\sigma_1,\ldots,\sigma_8 \in SoS$$

To express this problem as a semi-definite program or SDP (which can be solved with commercial solvers), all SoS constraints must be linear functions of the decision variable polynomials. However, Constraint 3 in the above problem includes the terms $\mathcal{L}^{\hat{x}}_{\hat{g}_p}\hat{v}\hat{u}_s$ and $\lambda\hat{v}$ which are *bilinear* in $\hat{u}_s$, $\hat{v}$ and in λ, $\hat{v}$ respectively. Problems of this form are referred to as bilinear sums-of-squares problems. The bilinear nature of the constraints means that these problems are non-convex, and we can no longer guarantee a globally optimal solution to this problem.

To solve this nonconvex bilinear sums-of-squares program we turn to a strategy called alternation. This strategy breaks (5.19) into a pair of linear sums-of-squares programs which can each be solved using a commercial solver. In each program one of the bilinear variables is kept fixed while the other is optimized over. The variables that were optimized are then fixed while the other pair of variables are optimized. If the final solution satisfies the constraints of the original program, the solution is guaranteed to be a viability domain. Computationally, each SDP is formulated in spotless[1] and solved using Mosek.

5.4.3 Guaranteed Safe Semi-autonomous Controller

We use a feasible solution to the above optimization problem to generate a guaranteed safe semi-autonomous controller. This controller modifies user input to ensure that the Viability Conditions 3 and 4 are always satisfied. Condition 4 can be enforced by saturating the user inputs to always lie within the input bounds. To enforce condition 3, we note that it is only active on the boundary of $\hat{V}$. This means that we can ensure safety so long as we use the autonomous safe controller $\hat{u}_s$ when the state lies on the boundary of $\hat{V}$, i.e. $\{\hat{x} \in Z | \hat{v}(\hat{x}) = 0\}$.

Since a controller that is discontinuous on the boundary of the safe set would pose difficulties for systems with finite bandwidth, we additionally must ensure that the new controller is continuous near the boundary. To achieve this, we smoothly interpolate between the user input $\hat{u}_0$ and the guaranteed safe controller $\hat{u}_s$ (which

[1] `https://github.com/spot-toolbox/spotless`

satisfies the safety condition when $\hat{v}(\hat{x}) = 0$) to get the regulated input $\hat{u}_r$:

$$\hat{u}_r = \hat{u}_0 + (\hat{u}_s(\hat{x}) - \hat{u}_0) w_s(\hat{v}(\hat{x}), \epsilon), \tag{5.20}$$

where $\hat{u}_s$ and $\hat{v}$ are computed using (5.19), $w_s : \mathbb{R} \to [0, 1]$ is a smooth step-like function that satisfies $w_s(v, \epsilon) = 0$, $\forall v \geq \epsilon$, and $w_s(v, \epsilon) = 0$, $\forall v \leq \epsilon/2$, and $\epsilon \in (0, 1)$ controls the smoothness of the interpolation.

When $\hat{x}$ satisfies $\hat{v}(\hat{x}) > \epsilon$, the user input is unmodified, as we are sufficiently removed from the boundary of the safe set. When $0 \leq \hat{v}(\hat{x}) \leq \epsilon/2$, the safe controller is fully active, keeping the state in the safe set.

5.5 Results

We used the proposed approach to compute a viability domain for the robot Rabbit [18]. The viability domain is represented using a set of 8 degree-4 polynomials, each covering an interval within the full range of θ. A two-dimensional slice of the viability domain $\hat{V}$ is shown in Fig. 5.2.

To demonstrate the semi-autonomous safe controller, we used it to ensure safety while performing a reference following task. The task is to track a time-varying pitch angle $\alpha_d : [0, \infty) \to A$. To follow the target, we set a desired pitch acceleration u_α using a "naïve" PD controller:

$$u_\alpha^d(x_\alpha, t) = k_p(\alpha_d(t) - \alpha) + k_d(\dot{\alpha_d}(t) - \dot{\alpha}) + \ddot{\alpha_d}(t). \tag{5.21}$$

We used the feedback controller (5.8) to map the desired acceleration to the four motor torques of the Rabbit model.

For the feedback controller to respect Rabbit's actuator torque limits, we first saturated u_α with a real-time Quadratic Program (QP) to get the input to our safety regulator:

$$\begin{aligned} \hat{u}_0(x, x_\alpha, t) = &\min_{u_\alpha} \left| u_\alpha - u_\alpha^d(x_\alpha, t) \right|^2 \\ &\text{s.t. } u^*(x, x_\alpha, u_\alpha) \in [-30\,\text{Nm}, 30\,\text{Nm}]^4 \end{aligned} \tag{5.22}$$

Using a QP to satisfy the actuator constraints of the system is similar to many state of the art approaches for high-dimensional robot control [41, 66, 68, 65] . A major limitation of these approaches is the inability to guarantee the feasibility of the QP. That is, for some states, there may not be an input that satisfies the actuator

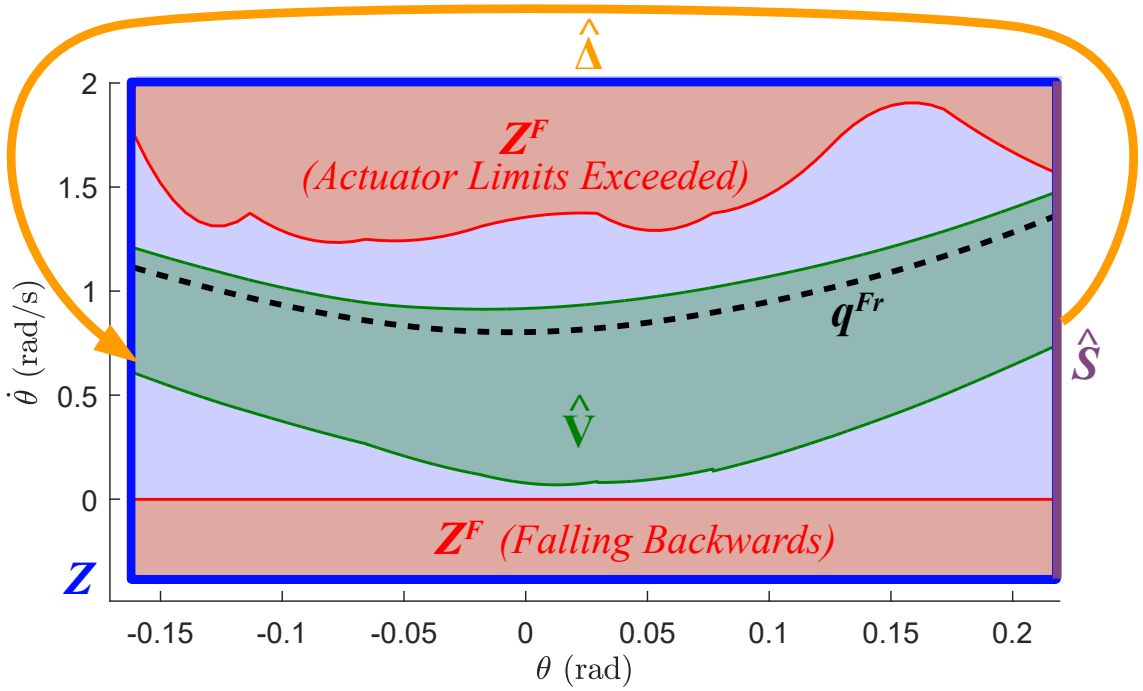

Figure 5.2: A 2D slice (along $\alpha = \dot{\alpha} = 0$) of the four-dimensional viability domain $\hat{V}$ (shown in green) for Rabbit. The border at the right corresponds to the hybrid guard $\hat{S}$ of foot touchdown, where the state is reset under the map $\hat{\Delta}$ to the left of the figure. The unsafe set Z^F is shown in red. We avoid the lower region ($\dot{\theta} < 0$) in order to conservatively prevent backwards falls. The upper region conservatively approximates the region in which the control input (5.8) violates the torque limits of the robot. By modifying the control input whenever Rabbit is at the edge of $\hat{V}$, Z^F can be avoided indefinitely. Finally, the periodic trajectory used to generate our targets q^{Fr} is shown in dashed black. Note that our viability domain is able to guarantee robot safety even for states far away from this nominal trajectory.

constraints (the set of such states is shown in red in Fig. 5.2).

Our approach guarantees the feasibility of (5.22) by constraining the state of the robot to be within the QP-feasible region (i.e. outside of Z^F in Fig. 5.2). To maintain this state constraint, we modified the input $\hat{u}_0$ using the guaranteed safe semi-autonomous controller defined in (5.20). In Fig. 5.3, we compare the results of the naïve controller (5.21) and the safe controller (5.20) using a simulation of the full dynamics of the robot Rabbit.

When tracking the backwards pitch target, the naïve controller slows to the point of falling backwards, while the safe controller deviates slightly to maintaint forward walking. For the forward pitch target, the naïve controller speeds up as it leans forward. At a certain speed, it cannot longer stay on the low dimensional manifold under the torque limits and falls. The safe controller recognizes this risk early and deviates from the desired forward pitch before reaching this speed. The bottom-left figure shows how the set of torque-limit satisfying control inputs disappears for the naïve controller.

This task demonstrates that robot safety can be maintained even for states that are far away from any periodic limit cycle. Indeed, the only periodic limit cycle used in our approach keeps the body pitch relatively upright ($\alpha = 0$). As such, our approach broadens the set of real-time safe behaviours that can be executed by Rabbit, since previous methods [4, 66, 65, 63] would all require a pre-computed limit cycle for each new reference trajectory.

5.6 Conclusion

This chapter presents a method to construct a guaranteed safe semi-autonomous controller for high-dimensional walking robots. The resulting controller guarantees viability and allows for flexible input when viability is not at risk. The method is evaluated on a model of the robot Rabbit, and a tracking task is used to illustrate its capabilities. With a 10-dimensional state space, this model is larger than any known model for which safety guarantees have been generated.

Despite this increase in model dimension, our example is still somewhat simplified: the dynamics are two dimensional, the terrain is flat, and the range of behaviour is limited to modifying the torso pitch angle. In contrast, bipedal robots in the world must traverse three dimensional, varied terrain while performing a wide range of tasks.

When extending our method to these cases, a trade-off arises between the degree

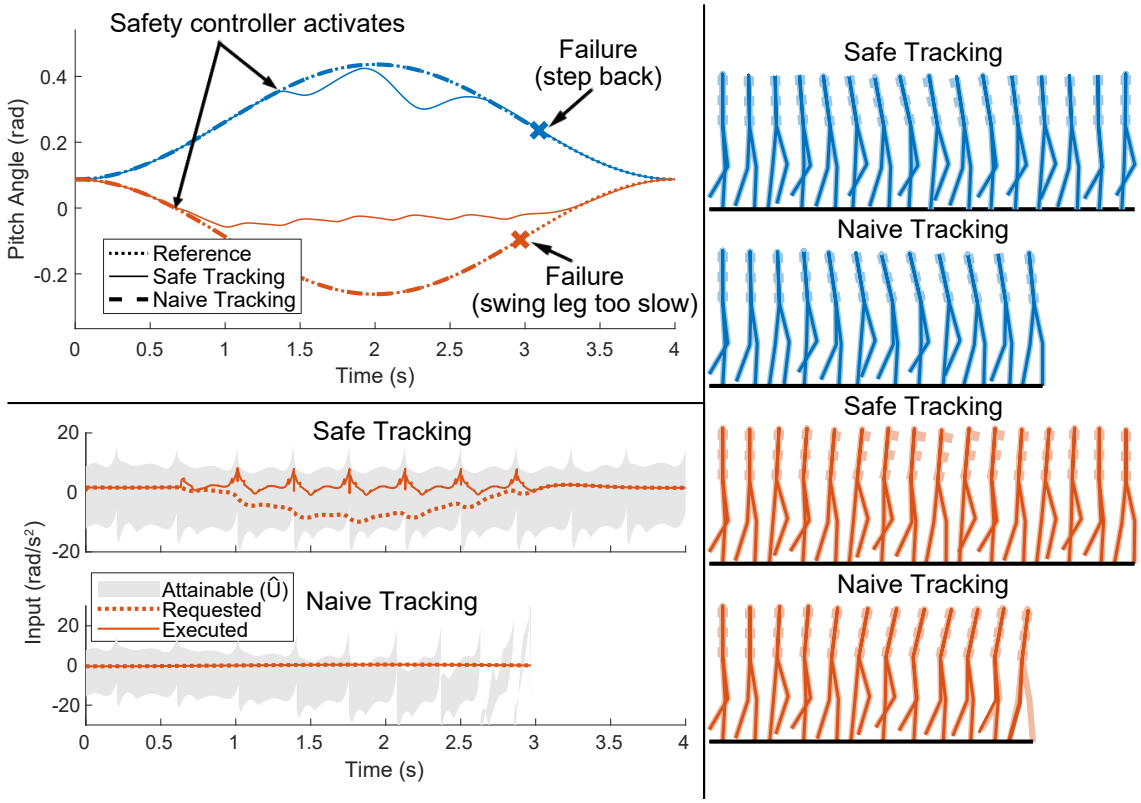

Figure 5.3: Tracking performance of the safe (5.20) and naïve (5.21) controllers following two reference trajectories under the full rabbit dynamics. The pitch angles are shown in the top left. For both references, the safe controller modifies the input before safety is at risk, while the naïve controller follows the reference even as it leads to failure. Failure for the upper trajectory corresponds to stepping backwards, and in the lower trajectory corresponds to moving too fast for the swing leg to reach its target. The bottom left figure shows desired input u_α^d and executed input for both naïive and safe tracking controllers following the second reference target. The state-dependent region of inputs that satisfy the torque constraints are shown in grey. Note that under the naïve controller, this region vanishes as the forward walking speed of the robot becomes too high. Stills from the simulation trajectories are shown on the right. The dotted line is the desired pitch, and the faded line is the nearest on-manifold state $q_0(\theta, \alpha)$.

of underactuation of the model, the genericity of the behaviour (i.e. the number of shaping parameters), and the computational complexity of the optimization problem. From Section 5.3.2, the dimension of the reduced order manifold (our state space) is twice the sum of the degree of underaction and the number of shaping parameters. In [71], the authors show that a 6 dimensional state space is tractable for similar sums-of-squares programs. Our approach can thus currently handle a maximum of three degrees of underactuation and/or shaping parameters.

Under this constraint, we can directly extend our method to 3D. For instance, take the 3D biped with controlled steering given in [63]. This application has two degrees of underactuation (pitch and yaw) and would have one shaping parameter controlling yaw rate (i.e. turning left or right). Using our method, we could construct a safe steering controller for the robot that avoids the risk of turning too quickly and falling. An extension to rough terrain, however, will likely require improvements in scaling of the sums-of-squares problem. Such scaling improvements are an active research target [69, 1].

The core insight behind our approach is that sums-of-squares and hybrid zero dynamics are remarkably complementary tools. Sums-of-squares analysis generates the set based guarantees needed to render hybrid zero dynamics safe, and hybrid zero dynamics provides the dimensionality reduction needed for sums-of-squares analysis to be tractable. The key innovation for combining these two tools was the introduction of a set of shaping parameters which control the dynamics on the manifold. The ability to combine sums-of-squares and hybrid zero dynamics presents a promising path forward for building guaranteed safe walking controllers for complex legged robots.

CHAPTER VI

Stepping Onto Rough Terrain: Reachability-based Gait Selection

This chapter continues moving our safety regulation approach towards more realistic walking robot scenarios, namely walking over rough terrain. At the time of this dissertation, the work presented here is still in progress. As a result, this section is somewhat speculative: we propose our method in detail and discuss the limitations, but have not yet generated results.

6.1 Introduction

In the previous chapter, we demonstrate how a 2D robot moving on flat terrain can actively modify its torso angle in a guaranteed safe manner. This single behavioural modification is however not sufficient to deal with uneven terrain or physical obstacles as would be encountered in real-world environments. The first step towards practical application of our approach must therefore be to increase the available selection of guaranteed safe behaviours. This increased range of behaviours should include, for example, stopping, acceleration/deceleration, and varied step length and step height.

In this chapter, we propose *Reachabillity-based Gait Selection* (RGS): a control methodology that allows for an increased range of guaranteed safe behaviours. This methodology is inspired by the *Reachability-based Trajectory Design* (RTD) approach that is being developed for the safe control of autonomous vehicles [52, 100, 51]. In the RTD approach, an outer approximation of the forward reachable set is computed (using sums-of-squares) for a parametrized family of vehicle trajectories. This forward reachable set is then intersected against potential obstacles in order to find the set of safe trajectories. In real-time, the controller can then freely select between this set of safe trajectories in order to achieve a higher-level objective (e.g. a lane change).

If no safe trajectory exists, the controller switches to a stopping controller, which is guaranteed to safely stop the vehicle without crashing.

RGS takes a similar approach. We first construct a standing controller, which stabilizes the robot around a stationary standing position, and find an inner approximation to its region of attraction (ROA, the set of states which can be stabilized under the standing controller). We then construct a family of two-step walking gaits, along with their forward reachable sets (FRS, the set of possible outcome states after the robot takes two steps). In real-time, the controller then selects between the walking trajectories whose forward reachable set is contained within the region of attraction of the stopping controller. This guarantees that the robot can always safely come to a stop while selecting between behaviours.

When performing a sums-of-squares analysis over a large range of robot behaviours (e.g. when computing forward reachable sets), the curse of dimensionality quickly becomes a major limitation. If we were to parametrize each class of behavior with a continuously adjustable *shaping parameter* as in the last chapter, we would quickly exceed the 6 dimensional capability of our sums-of-squares analysis. Indeed, as mentioned in Section 5.6, this approach would limit a control designer to a maximum of 2 behavioural modifiers for a robot with one degree of underactuation. However, to compute our FRS for the two-step walking controller, we would need 4 behavioural modifiers, one each for the height and length of both the first and the second step. An alternative approach is therefore required.

We address this obstacle by decoupling the sums-of-squares analysis. Since finding a single forward reachable set for all possible behaviours is computationally demanding, we instead aim to break the analysis into two steps. In the first, we compute the FRS of just the first step of the walking controller. In the second we find the set of states that can be brought to a standstill in one step by computing the backwards reachable set (BRS, the set of states that can reach the standing controller ROA in one step) of our standing controller's ROA. Provided that the ending configuration of the first step is fixed to be the same as the starting configuration of the second step, we can then freely mix and match the parameters of the first and second steps. We can then ensure that the robot can stop in two steps by finding first and second step parameters such that the FRS of the first step is within the second step BRS of the standing ROA. This idea of composing multiple intersecting gaits to allow for safe switching between behaviours bears strong similarity to the approach used in [63, 101, 102].

Throughout this chapter we use a Rabbit model walking over rough terrain to

illustrate the RGS approach (see Figure 6.1). This example bears strong resemblance to the stepping stones example given in [64]. Our approach differs from this work, in that it provides a guarantee that the velocity of the underactuated degree of freedom of the robot remains within limits, even on very steep terrain.

Note that this chapter does not present a working implementation of the method, which remains a work in progress at the time of this dissertation. Additionally, note that this chapter uses the same notation introduced in Chapter V.

6.2 Reachability-based Gait Selection

An overview of the RGS approach for a Rabbit model walking over rough terrain is given in Figure 6.1. The remainder of this section contains a discussion of the standing controller and its region of attraction (Section 6.2.1), a discussion of the mid-stance configuration (Section 6.2.2), a description of the first step controller and the forward reachable set computation for this controller (Section 6.2.3), a description of the second step stopping controller and its backwards reachable set to the standing ROA (Section 6.2.4), and finally a description of the gait selector (Section 6.2.5).

6.2.1 Standing Controller and Region of Attraction

The goal of the standing controller is to stabilizes the robot around a stationary standing position. Since Rabbit does not have ankles, the *standing controller* uses the torso angle to stabilize the robot about a standing position. If we fix the swing leg and only control the torso angle, the resulting robot is dynamically equivalent to an *acrobot* [91]. Stabilizing the upright position of the acrobot is a well known robotics problem which has many approaches. We chose to use the sums-of-squares approach given in [55], as it additionally generates an inner approximation of the region of attraction of the standing controller. The result of this process is a standing configuration q_s and a set of initial hip and pitch velocities which can be brought to stable standing.

6.2.2 Mid-stance Configuration

At each mid-stance the controller switches discretely between gait parameters. In order to smoothly execute this discrete transition in gait parameters, we require every member of the family of walking gaits to share a common mid-stance configuration q_m and configuration velocity $\frac{\partial q_m}{\partial \theta}$. Note that we allow the mid-stance joint velocities $\dot{q}$

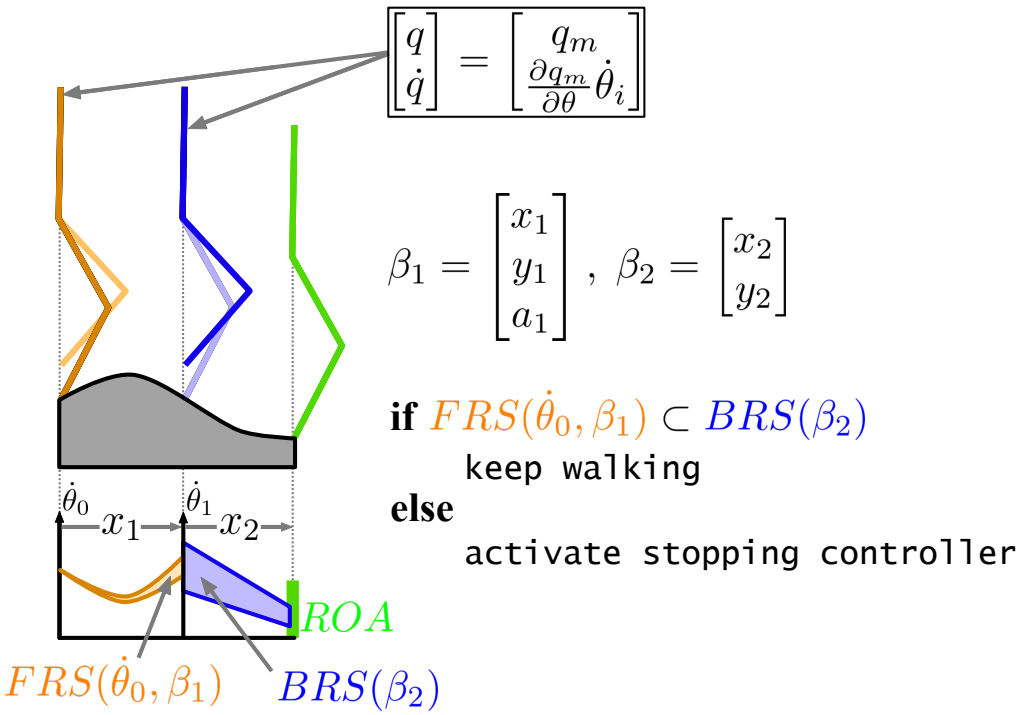

Figure 6.1: Overview of the reachability-based gait selection approach for a Rabbit model walking over rough terrain. The approach begins by constructing a standing controller that stabilizes the robot about a stationary standing position. Next, we specify a mid-stance configuration q_m and configuration velocity $\frac{\partial q_m}{\partial \theta}$. We then generate a parametrized family of first steps (using interpolated FROST trajectories) that start and end at the mid-stance configuration for a range of step-lengths (x_1), step heights (y_1), and accelerations (a). Next we construct a stopping controller (using the methods of the previous chapter) that brings the robot to a stop in one step (of step-length x_2 and step-height y_2) by actively modifying the pitch angle. Finally, we compute three set-based objects: a *region of attraction* (ROA) of the standing controller, a *forward reachable set* (FRS) of the first-step controller, and a *backwards reachable set* of the ROA using the second-step stopping controller. The FRS takes in the current mid-stance velocity ($\dot{\theta}_0$) and the discrete parameters of the first step (β_1), and returns a bound on the next mid-stance velocity ($\dot{\theta}_1$). The BRS takes in the parameters of the stopping step (β_2), and returns the set of mid-stance velocities that can safely be brought to a stop in one step. At each mid-stance event, the gait-selector searches for discrete step parameters (β_1, β_2) that can bring the robot to a stop in two steps (i.e. $FRS(\dot{\theta}_0, \beta_1) \subset BRS(\beta_2)$). If the selector succeeds, the robot takes the first step using parameters values β_1, and the selection process repeats at the next mid-stance. If the selector fails, the robot executes a stopping step using the parameter values β_2 from the previous mid-stance. Since the previous selection step ensured that the robot state is within the BRS of the stopping controller, we know that the robot can safely be brought to a stop.

to scale linearly with the hip angle velocity at mid-stance $\dot{\theta}$. This constraint ensures that the hybrid zero dynamics manifolds of each stepping controller gait intersect smoothly at mid-stance.

The choice of mid-stance configuration is important, since the robot must pass through this configuration at every step. If poorly chosen, this might limit the available behaviour of the robot. For instance, if the swing leg configuration is too far behind the hip, the maximum walking speed might be reduced, since the leg can only be brought forward after mid-stance has been reached.

It is likely that a metric exists over which this configuration should be optimized. The form of such a metric, however, is still uncertain at the time of this dissertation. We therefore suggest using a heuristic choice of mid-stance configuration, that can be modified as limitations are encountered.

6.2.3 Stepping Controller and Forward Reachable Set

Once a mid-stance configuration has been selected, the next task is to construct a stepping controller that moves the robot from one midstance to the next. Since the robot is walking over rough terrain, this stepping controller must be able to accommodate a continuous range of step-lengths (x_1) and step-heights (y_1). Additionally, in order to be able to regulate speed, this controller should be able to speed up and slow down the robot using a desired acceleration ($a_1 = \frac{\dot{\theta}_1}{\dot{\theta}_0}$).

One example of a controller that can accommodate all of this variation is a hybrid zero dynamics gait-library [21, 31]. We use this idea to construct a 3 dimensional gait library parametrized by the discrete gait parameters: $\beta_1 = [x_1, y_1, a_1]^\top$. We briefly describe the construction of the gait library here.

First, a series of trajectory optimizations are conducted over a coarse grid (approximately $10 \times 10 \times 10$) of gait parameters. Each trajectory optimization searches for a robot trajectory which satisfies the desired midstance configuration, step length, step height, and acceleration, as well as a fixed midstance velocity. In order to promote smoothness of the resulting trajectories, a term is added to the cost function which penalizes deviation from nearby trajectories. As in Example 3 from the previous chapter, we fit a phasing function to find a phase-dependent configuration target for each trajectory.

Next a second-order-smooth interpolation is conducted over the grid of configuration targets to obtain a single configuration function ($q^{Fr}(\theta, \beta_1)$) that is dependent on stance leg angle as well as gait parameter. The stepping controller then smoothly tracks this configuration target using a PD position controller with feed-

forward torques at each joint.

In order to safely bound the behaviour of the stepping controller when selecting gaits, we must construct an outer approximation of the forward reachable set ($FRS : \mathbb{R}^4 \rightarrow 2^{\mathbb{R}}$). FRS is a set-valued function which takes as input the hip velocity at midstance ($\dot{\theta}_0$) as well as the step parameters of the next step (β_1), and returns the set of possible next-step mid-stance velocities ($\dot{\theta}_1$). Note that for some combinations of mid-stance velocity and step-parameter, the robot will fail before the next mid-stance. We assign the value -1 to any failed trajectory (since stance hip position is monotonic, a negative midstance velocity at the next step is impossible) when constructing this set. We generate an outer approximation of this set using the functions $FRS_{min} : \mathbb{R}^4 \rightarrow \mathbb{R}$ and $FRS_{max} : \mathbb{R}^4 \rightarrow \mathbb{R}$, which satisfy:

$$FRS(\dot{\theta}_0, \beta_1) \subset [FRS_{min}(\dot{\theta}_0, \beta_1), FRS_{max}(\dot{\theta}_0, \beta_1)] \tag{6.1}$$

One approach to find these bounding functions, is to use sampling. First, a uniform 4D grid of initial velocities and step parameters is generated and simulated forward to the next midstance using an ODE solver. Next, a linear program is run to find the smallest upper bounding and largest lower bounding polynomial functions, i.e. FRS_{min} and FRS_{max}.

Another approach to find these functions uses methods similar to those of Chapter V. First a reduced order hybrid zero dynamics manifold is created for the set of discrete parameters $\dot{\theta}_0$ and β_1 using the same methods as in Sec 5.3. Note that since $\dot{\theta}_0$ and β_1 are stationary parameters, we don't need to include their velocities in the HZD manifold. The 5D state of the resulting manifold is given by:

$$\hat{x}_{FRS} = \begin{bmatrix} \theta \\ \dot{\theta} \\ \dot{\theta}_0 \\ \beta 1 \end{bmatrix} \tag{6.2}$$

Once this manifold is constructed, either sums-of-squares [52] or zonotopes [25] are used to compute an outer approximation of the forward reachable set. In this computation, the initial set is given by the plane $\{\hat{x}_{FRS} | \theta = 0, \dot{\theta} = \dot{\theta}_0\}$. The resulting set can be translated into an outer approximating interval using a simple linear program.

6.2.4 Stopping Controller and Backwards Reachable Set

The goal of the second step stopping controller is to bring the robot from the midstance configuration into the region of attraction of the standing controller in one step. Additionally, we aim to find an inner approximation to the backwards reachable set ($BRS : \mathbb{R}^2 \to 2^{\mathbb{R}}$) of the stopping controller. The backwards reachable set takes as input a step length and step height, and returns all midstance velocities that can be brought to a standstill in one step.

The stopping controller bears strong similarities to the stepping controller in Section 6.2.3. We begin by using FROST to generate a series of trajectories that begin at q_m and ends at q_s for various step lengths (x_2) and step heights (y_2). This set of step parameters is collected into the discrete stopping parameters $\beta_2 = [x_2, y_2]^\top$. As in the stepping controller, we use a second-order-smooth interpolation to generate a hybrid zero dynamics manifold parametrized by β_2.

Next, in order to add a degree of control to the manifold, we use the methods of the previous chapter to add a continuous shaping parameter α_2. This parameter modifies the pitch angle continuously as in Section 5.3. The resulting 6D controlled hybrid zero dynamics manifold is parametrized by the state:

$$\hat{x}_{BRS} = \begin{bmatrix} \theta \\ \dot{\theta} \\ \beta_2 \\ \alpha_2 \\ \dot{\alpha}_2 \end{bmatrix} \tag{6.3}$$

With this manifold in hand, we can use sums-of-squares analysis [56] to find a feedback control law that guides the robot towards the region of attraction of the standing controller. The sums-of-squares analysis also generates an inner approximation ofthe backwards reachable set of this region of attraction. This inner approximation is given as the zero super-levelset of a polynomial $w : \mathbb{R}^3 \to \mathbb{R}$. Any midstance velocity $\dot{\theta}_1$ and step parameters β_2 satisfying $w(\dot{\theta}_1, \beta_2) > 0$ are in the backwards reachable set of the standing controller. That is w satisfies:

$$w(\dot{\theta}_1, \beta_2) < 0, \ \forall \dot{\theta}_1 \in BRS(\beta_2) \tag{6.4}$$

We additionally restrict the sums-of-squares to only find a convex backwards reachable set by limiting the parametrization of w. By forcing the backwards reach-

able set to be convex, we can verify the relation $FRS(\dot{\theta}_0, \beta_1) \subset BRS(\beta_2)$ by simply checking that the endpoints of our FRS interval are within the convex inner approximation of $BRS(\beta_2)$. That is, we check that: $w(FRS_{min}(\dot{\theta}_0, \beta_1), \beta_2) > 0$ and $w(FRS_{max}(\dot{\theta}_0, \beta_1), \beta_2) > 0$.

6.2.5 Gait Selector

The gait selector is the routine that selects between the gait parameters in order to keep the robot from falling over. At each mid-stance event, the gait-selector searches for discrete step parameters (β_1, β_2) that can bring the robot to a stop in two steps (i.e. that satisfy $FRS(\dot{\theta}_0, \beta_1) \subset BRS(\beta_2)$). Since there are likely many parameters satisfying this constraint, the selector may also minimize some cost function in its choice. If the selector succeeds in finding parameters that satisfy the safety constraint, the robot takes the first step using parameters values β_1, and the selection process repeats at the next mid-stance. If the selector fails, the robot executes a stopping step using the parameter values β_2 from the previous mid-stance. Since the previous selection step ensured that the robot state is within the BRS of the stopping controller, we know that the robot can safely be brought to a stop.

Note that since we use a one-step stopping controller, and a one-step walking controller, the selector has the ability to plan over a two step horizon. This is motivated by observations from simple models [110], and from human experiments [58], which indicate that planning two steps ahead is sufficient for stable walking over rough terrain.

6.3 Discussion and Limitations

In this chapter, we proposed reachability-based gait selection: a control approach that allows for a range of robot behaviour while explicitly ensuring the robot will not fall over. We additionally propose an implementation of our approach that would allow a Rabbit model to walk continuously over rough terrain without risk of falling.

While the RGS approach adds significant capability to the methods presented in Chapter V, there is still much ground left to cover in moving towards a real-world implementation. Remaining challenges include managing disturbance, model uncertainty, and sensor noise, as well as extending to 3D.

CHAPTER VII

Conclusions and Future Directions

In this dissertation, we have not made a robot that moves more efficiently than Ranger [11], nor have we made a robot that moves more robustly than Atlas [24]. Instead, we have provided fundamental and novel approaches that can improve these goals in principle. By doing so, our work can be directly applied to a wide range of current and future robots.

These contributions are five-fold. The first points in the direction of energetic economy, where we find the ideal energetically economical motions for a detailed model of the robot RAM*one* in Chapter II. Next, we pivot this result towards robustness in Chapter III, where we conduct an investigation into the energetic economy of walking motions for the physical robot RAM*one*. In this chapter, we also encounter an explicit trade-off between economy and robot safety. This trade-off motivates a shift in our investigation towards robot safety as a constraint. The motivation behind this shift is that enforcing safety as a constraint allows robot controllers to freely maximize robot economy without risk of falling. In Chapter IV, we formally state this safety constraint through the lens of viability analysis, and show how it can be constructed and enforced for an actuated compass-gait walker. In order to scale this approach to more realistic robot models, Chapter V presents a dimension-reducing technique for our safety analysis, and shows how it can be used to generate a safety constraint for a full model of the robot Rabbit. Finally, in Chapter VI, we propose a path towards constructing safety constraints for a wider variety robot behaviour, such as walking over rough terrain.

While several advances have been made in the course of this investigation, there is significant ground left to cover. The aim of this chapter is to begin mapping out the remaining ground. We first frame our contributions in their broader context in Section 7.1. We then lay out three avenues for future development in Section 7.2. Finally, we leave the reader with some concluding remarks.

7.1 Discussion of Contributions

At its widest scope, this dissertation will help researchers analyze and improve the energetic economy and robustness against falling of bipedal robots. Sufficient improvements in these qualities will grant bipedal robots access to a host of previously inaccessible domains. One potential application of this increased access is in search and rescue robots that can assist in the location and retrieval of people stuck in rugged, forested, and mountainous terrain. Other applications include home care robots that can move from one floor to another or leave the house to pick up groceries, delivery robots that can cover the last mile from distribution center to home, and patrol robots that monitor construction sites or oil rigs.

More narrowly, each chapter of this work provides its own contribution to the field of locomotion research.

By drawing parallels between robotic and biological gait, Chapter II provides new insight for biologists and roboticists alike. From a biomechanical perspective, the energetic optimality of biological templates for a detailed robotic system strengthens the hypothesis that these templates are chosen as energetic minimizers for biological systems. From a robotics perspective, the discrete difference between energetically optimal motions at low speeds and high speeds suggests that locomotion controllers could benefit from a discrete switch from walking to running as speed increases.

The impact of Chapter III is twofold. First, it strengthens the impact of the previous aim by validating the results against reality. Second, it provides a hardware demonstration of two principles of energetically economical locomotion: that bipeds can derive benefit from straightening their legs while walking, and that economically desirable motions can lie at the boundary of failure. These principles provide valuable guidance when constructing future walking controllers.

Chapters IV-VI provide an approach that can be used by control designers to conduct formal safety analysis and control synthesis for complex walking robot models. This approach will help move the question of walking robot safety away from heuristic guesses and towards understandable guarantees with explicit assumptions. Having such guarantees can free a control engineer to design for other tasks without having to consider the implications on robot safety. These explicit guarantees also open the door for learning based methods that would otherwise be too dangerous to use on expensive robot hardware.

Figure 7.1: The bipedal robot Cassie (left) and a realistic model (right). Image taken from the video *Cassie Sim Comparison* released by Agility Robotics [80].

7.2 Future Directions

The ultimate aim of this work is to improve the robustness and energetic economy of real walking robots in order to enable safe and efficient mobility across the wide variety of terrain that is readily accessible on foot. Robot hardware is currently being built that has the capacity for this mobility, including Agility Robotics' Cassie [79], and Boston Dynamics' Atlas [23]. The control of these robots unfortunately still comes up short, especially when compared to human mobility. These robots fall often, even on smooth and flat terrain, and each new behaviour must be tested and tuned extensively (with lots of falls involved in the process).

There is thus clear value in generating safety constraints for robots like Cassie and Atlas, and we believe our tools can be extended towards this end. To get there, two main extensions must be completed: the extension from 2 to 3 dimensions, and the extension from simulation to reality. Additionally, once safety constraints have been extended to reality, we can then safely conduct studies into the energetic economy of real-world robots using guaranteed-safe online learning. We discuss each of these three extensions briefly below.

7.2.1 Safety in 3 Dimensions

In order to find safety guarantees for real robots, we must first extend the results of Chapters IV-VI from two to three dimensions. This extension will likely begin in simulation, using a model similar to the Cassie model in Figure 7.1.

One approach to this is to directly apply the methods of Chapter V, using two degrees of underactuation in the Hybrid Zero Dynamics manifold: one for robot pitch, and one for robot roll. If we include a single shaping parameter, e.g. controlling the turning, we would have a six-dimensional sums-of-squares problem, which is at the limit of computational feasibilty, but likely achievable (a 6-dimensional bilinear sums-of-squares problem is solved in [71]). Further improvements may be made possible by removing the lateral degree of underactuation from the manifold. To preserve safety, we can use sampling to bound the effects of the lateral dynamics on sagittal motion then treat the result as a disturbance in the sums-of-squares optimization.

7.2.2 From Model to Reality

The safety methods in this dissertation rely heavily on robot models with bounded uncertainty in their dynamics. For our safety guarantees to extend to hardware, we must first ensure that the model conservatively captures the behaviour of the physical robot. This can be framed as a *simulation relation* [20], in that at least one model trajectory within the bounded uncertainty must be able to exactly match each hardware trajectory.

If we are able to densely sample the robot dynamics throughout its state-space, such a relation can be obtain by fitting an uncertainty bound between the measured and modeled robot trajectories. A similar approach has previously been shown to successfully bound hardware behaviour for autonomous vehicles [100]. The high state-space dimension and multiple degrees of underactuation of walking robots will make them a more challenging target.

7.2.3 Guaranteed Safe Online Learning

Online learning is a valuable tool for achieving high performance behaviour in physical systems when modelling accuracy is limited. This is particularly true for legged robots due to the inherent difficulty of accurately modelling contact events. However, such learning schemes are traditionally challenging for legged robots due to the high cost of falling (which can require lengthy hardware repairs). As such, successful learning implementations on walking robots have been largely limited to hardware in which either the likelihood or the cost of falling is low [97, 48].

We see the safety constraints in this dissertation as a promising way to mitigate this risk. By removing falling from the robot operation, we can safely explore the landscape of walking behaviours to maximize a wide variety of performance metrics.

7.3 Concluding Remarks

During the 5 year course of this dissertation, the field of legged robots has made enormous strides. Walking robots have begun to move into real-world environments, promising to deliver packages [103], and monitor construction sites [36]. This push towards commercialization comes with a new set of challenges. Heuristic, single-task controllers that work well in lab environments must be generalized to handle a wider variety of situations and environments. Robots must be able to reason about their environment in real time as challenges arise, and construct safe and efficient strategies to overcome these challenges. I strongly believe that methods based on modelling and optimization, such as those presented in this dissertation, will remain to be vital tools for achieving this generalization.

APPENDIX

APPENDIX A

Derivation of Hybrid Invariance for the Controlled HZD Manifold

In this appendix, we first expand on the *hybrid invariance condition* of Section 5.3. We next use the expanded condition to construct an analytic representation of the modifying term h_m from (5.13). This term guarantees that the outputs used in our example satisfy the *hybrid invariance condition*.

A.1 Expanded Hybrid Invariance Condition

In this section, we translate the set based *hybrid invariance condition* of Section 5.3 into a set of equivalent conditions on the target functions. We will construct a set of targets that meet these conditions in the next section.

We begin by observing that since the outputs of Section 5.3 are chosen to satisfy hypothesis HH 3 from [104], we know that the map $\Phi : Q \times A \to \mathbb{R}^{n_q+n_\alpha}$ given by $\Phi(q, \alpha) := [h(q, \alpha), \theta(q), \alpha]^\top$ is a diffeomorphism onto its image.

This map helps us define a coordinate transform from the manifold parameters to the ambient space $q_0 : Z \to Q$ as:

$$\begin{bmatrix} q_0(\theta, \alpha) \\ \alpha \end{bmatrix} := \Phi^{-1} \begin{bmatrix} [0, \dots, 0]^T \\ \theta \\ \alpha \end{bmatrix}, \tag{A.1}$$

The output of this transform: $q_0(\theta, \alpha)$ is the full robot configuration associated with the manifold state at coordinates: $(\theta, \dot{\theta}, \alpha, \dot{\alpha}) \in Z$.

For the zero dynamics manifold to be forward invariant for the hybrid system, it must map onto itself through the guard and reset. This gives us the *hybrid invariance condition* (5.7) which Z must satisfy.

That is, provided that the robot state is on Z immediately before the impact:

$$\begin{bmatrix} q^-(\theta^-, \alpha^-) \\ \dot{q}^-(\theta^-, \alpha^-, \dot{\theta}^-, \dot{\alpha}^-) \end{bmatrix} = \begin{bmatrix} q_0(\theta^-, \alpha^-) \\ \frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-)\dot{\theta}^- + \frac{\partial q_0}{\partial \alpha}(\theta^-, \alpha^-)\dot{\alpha}^- \end{bmatrix}$$

then the post-impact state must also be on Z (where the superscripts $^-$ and $^+$ are used to indicate states immediately before and after the impact respectively). This gives an alternate form of the *hybrid invariance condition*:

$$h(q^+, \alpha^+) = 0 \tag{A.2}$$

$$\mathcal{L}_f^x h(q^+, \dot{q}^+, \alpha^+) + \mathcal{L}_{f_\alpha}^{x_\alpha} h(q^+, \alpha^+, \dot{\alpha}^+) = 0 \tag{A.3}$$

Where the post-impact states are given by the reset map:

$$\begin{bmatrix} q^+(\theta^-, \alpha^-) \\ \dot{q}^+(\theta^-, \alpha^-, \dot{\theta}^-, \dot{\alpha}^-) \\ \alpha^+(\alpha^-) \\ \dot{\alpha}^+(\dot{\alpha}^-) \end{bmatrix} = \begin{bmatrix} \Delta_q q^-(\theta^-, \alpha^-) \\ \Delta_{\dot{q}}\left(q^-(\theta^-, \alpha^-)\right) \dot{q}^-(\theta^-, \alpha^-, \dot{\theta}^-, \dot{\alpha}^-) \\ \alpha^- \\ \dot{\alpha}^- \end{bmatrix}.$$

By substituting this post-impact state into (A.3), we obtain the following equivalent condition to (A.3):

$$\frac{\partial h}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}\left(q^-\right)\left(\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-)\dot{\theta}^- + \frac{\partial q_0}{\partial \alpha}(\theta^-, \alpha^-)\dot{\alpha}^-\right) + \frac{\partial h}{\partial \alpha}(q^+, \alpha^-)\dot{\alpha}^- = 0, \tag{A.4}$$

which must hold for all $(\theta^-, \dot{\theta}^-, \alpha^-, \dot{\alpha}^-) \in Z$ (for notation's sake we omitted the arguments of the q^+ and q^- terms).

In summary, any targets satisfying (A.2) and (A.4) define a hybrid zero dynamics manifold.

A.2 Hybrid Invariant Targets

With these expanded conditions in hand, we now construct the hybrid invariant targets (5.13) from Example 3 in Section 5.3. We re-write these targets as:

$$h(q,\alpha) = h_0(q,\alpha) + h_m(\theta(q),\alpha) = \begin{bmatrix} q_1 - q_1^{Fr}(t_\theta(\theta(q))) - \alpha \\ q_3 - q_3^{Fr}(t_\theta(\theta(q))) \\ q_4 - q_4^{Fr}(t_\theta(\theta(q))) + \alpha \\ q_5 - q_5^{Fr}(t_\theta(\theta(q))) \end{bmatrix} + h_m(\theta(q),\alpha) \quad \text{(A.5)}$$

The aim of this section is to construct the modifying term $h_m(\theta,\alpha)$.

In order to simplify the derivation, we begin by fixing the robot configuration immediately before impact: q^-. In this example, impact only happens when $\theta^- = \theta_{max}$, and the configuration is fixed according to the FROST trajectory at impact (along with a modified pitch angle) i.e.:

$$q^-(\alpha^-) = q_0(\theta^-,\alpha^-) = q^{Fr}(t_{max}) + \begin{bmatrix} \alpha^- \\ -\alpha^- \\ 0 \\ -\alpha^- \\ 0 \end{bmatrix} \quad \text{(A.6)}$$

Note that for the targets in (5.13), this impact configuration is preserved so long as

$$h_m(\theta_{max},\alpha) = 0,\ \forall\alpha. \quad \text{(A.7)}$$

We satisfy our first *hybrid invariance condition* (A.2) by fixing the post-impact configuration of the robot. Since the FROST trajectory is periodic, we have $q^{Fr}(0) = \Delta_q q^{Fr}(t_{max})$. Therefore, (A.2) is satisfied so long as

$$h_m(\theta_{min},\alpha) = 0,\ \forall\alpha. \quad \text{(A.8)}$$

This gives us the post-impact configuration as

$$q^+(\alpha^-) = q_0(\theta^+, \alpha^-) = q^{Fr}(0) + \begin{bmatrix} \alpha^- \\ -\alpha^- \\ 0 \\ -\alpha^- \\ 0 \end{bmatrix} \tag{A.9}$$

Next we look at the second *hybrid invariance condition* (A.4). Since this condition must hold for all $\dot{\theta}^-$ and for all $\dot{\alpha}^-$, we know that the following two equations must be satisfied:

$$\frac{\partial h}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}\left(q^-\right)\frac{\partial q_0}{\partial \alpha}(\theta^-, \alpha^-) + \frac{\partial h}{\partial \alpha}(q^+, \alpha^-) = 0 \tag{A.10}$$

$$\frac{\partial h}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}\left(q^-\right)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-) = 0 \tag{A.11}$$

Where we have dropped the arguments of $q^+(\alpha^-)$ and $q^-(\alpha^-)$.

From (A.6), we have $\frac{\partial q_0}{\partial \alpha}(\theta^-, \alpha^-) = [1, -1, 0, -1, -]^\top$. Note that this vector is perpendicular to the impact direction (since the swing foot position is unaffected by pitch angle). Since velocities perpendicular to the impact direction are unaffected by the impact impulse, we have the relation:

$$\Delta_{\dot{q}}\left(q^-\right)\frac{\partial q_0}{\partial \alpha}(\theta^-, \alpha^-) = \frac{\partial q_0}{\partial \alpha}(\theta^+, \alpha^-) = \frac{\partial q^+}{\partial \alpha}(\alpha^-) \tag{A.12}$$

Next, from our prior assumptions, it is clear that $h(q^+(\alpha^-), \alpha^-) = 0$. By taking the derivative with respect to α^- on both sides of this expression, we obtain:

$$\frac{\partial h}{\partial q}(q^+(\alpha^-), \alpha^-)\frac{\partial q^+}{\partial \alpha}(\alpha^-) + \frac{\partial h}{\partial \alpha}(q^+(\alpha^-), \alpha^-) = 0 \tag{A.13}$$

Now substituting (A.12) into (A.13), we see that condition (A.10) is satisfied.

To satisfy condition (A.11), we follow an approach similar to [92]. We begin by substituting (A.5) into (A.11) to obtain:

$$\left(\frac{\partial h_0}{\partial q}(q^+, \alpha^-) + \frac{\partial h_m}{\partial \theta}(\theta^+, \alpha^-)\frac{\partial \theta}{\partial q}(q^+, \alpha^-)\right)\Delta_{\dot{q}}\left(q^-\right)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-) = 0 \tag{A.14}$$

Re-arranging this we see that so long as $\frac{\partial \theta}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}\left(q^-\right)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-) \neq 0$, then condition (A.11) is satisfied so long as h_m satisfies:

$$\frac{\partial h_m}{\partial \theta}(\theta^+, \alpha^-) = -\frac{\frac{\partial h_0}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}(q^-)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-)}{\frac{\partial \theta}{\partial q}(q^+, \alpha^-)\Delta_{\dot{q}}(q^-)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-)} \tag{A.15}$$

Note that the term $\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha^-)$ can in general depend on the form of h_m. In order to remove this dependency, we force h_m to satisfy the constraint:

$$\frac{\partial h_m}{\partial \theta}(\theta_{max}, \alpha) = 0, \ \forall \alpha. \tag{A.16}$$

Putting this all together, we have shown that condition (A.11) (and by extension the *hybrid invariance conditions*) will be met, so long as h_m satisfies constraints (A.7), (A.8), (A.15) and (A.16). We satisfy these constraints by construction, chosing:

$$h_m(\theta, \alpha) = -p_m(\theta)\frac{\frac{\partial h_0}{\partial q}(q^+, \alpha)\Delta_{\dot{q}}(q^-)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha)}{\frac{\partial \theta}{\partial q}(q^+, \alpha)\Delta_{\dot{q}}(q^-)\frac{\partial q_0}{\partial \theta}(\theta^-, \alpha)}, \tag{A.17}$$

where $p_m : [\theta_{min}, \theta_{max}] \to \mathbb{R}$ is a cubic spline satisfying: $p_m(\theta_{min}) = p_m(\theta_{max}) = \frac{dp_m}{d\theta}(\theta_{max}) = 0$, $\frac{dp_m}{d\theta}(\theta_{min}) = 1$.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Ahmadi, A.A., Majumdar, A.: Dsos and sdsos optimization: Lp and socp-based alternatives to sum of squares optimization. In: 2014 48th annual conference on information sciences and systems (CISS), pp. 1–5. IEEE (2014)

[2] Alexander, R.M.: Three uses for springs in legged locomotion. The International Journal of Robotics Research **9**(2), 53–61 (1990)

[3] Alexander, R.M.: A model of bipedal locomotion on compliant legs. Philosophical Transactions of the Royal Society of London B: Biological Sciences **338**(1284), 189–198 (1992)

[4] Ames, A.D., Galloway, K., Sreenath, K., Grizzle, J.W.: Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. IEEE Transactions on Automatic Control **59**(4), 876–891 (2014)

[5] Ames, A.D., Tabuada, P., Jones, A., Ma, W.L., Rungger, M., Schürmann, B., Kolathaya, S., Grizzle, J.W.: First steps toward formal controller synthesis for bipedal robots with experimental implementation. Nonlinear Analysis: Hybrid Systems **25**, 155–173 (2017)

[6] Ames, A.D., Xu, X., Grizzle, J.W., Tabuada, P.: Control Barrier Function Based Quadratic Programs for Safety Critical Systems. IEEE Transactions on Automatic Control (2016)

[7] Ames, A.D., Zheng, H., Gregg, R.D., Sastry, S.: Is there life after Zeno? Taking executions past the breaking (Zeno) point. In: American Control Conference, 2006, pp. 6–pp. IEEE (2006)

[8] Aubin, J.P.: Viability theory. Springer Science & Business Media (2009)

[9] Bemporad, A.: Reference governor for constrained nonlinear systems. IEEE Transactions on Automatic Control **43**(3), 415–419 (1998)

[10] Bhounsule, P.A., Cortell, J., Grewal, A., Hendriksen, B., Karssen, J.D., Paul, C., Ruina, A.: Low-bandwidth reflex-based control for lower power walking: 65 km on a single battery charge. The International Journal of Robotics Research **33**(10), 1305–1321 (2014)

[11] Bhounsule, P.A., Cortell, J., Ruina, A.: Design and control of Ranger: an energy-efficient, dynamic walking robot. In: Proc. CLAWAR, pp. 441–448 (2012)

[12] Blickhan, R.: The spring-mass model for running and hopping. Journal of biomechanics **22**(11-12), 1217–1227 (1989)

[13] Bock, H.G., Plitt, K.: A multiple shooting algorithm for direct solution of optimal control problems. In: 9th IFAC World Congress, pp. 242–47. Budapest, Hungary (1985)

[14] Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge university press (2004)

[15] Burden, S.A., Gonzalez, H., Vasudevan, R., Bajcsy, R., Sastry, S.S.: Metrization and simulation of controlled hybrid systems. IEEE Transactions on Automatic Control **60**(9), 2307–2320 (2015)

[16] Cavagna, G.A., Heglund, N.C., Taylor, C.R.: Mechanical work in terrestrial locomotion: two basic mechanisms for minimizing energy expenditure. American Journal of Physiology-Regulatory, Integrative and Comparative Physiology **233**(5), R243–R261 (1977)

[17] Channon, P., Hopkins, S., Pham, D.: Derivation of optimal walking motions for a bipedal walking robot. Robotica **10**(02), 165–172 (1992)

[18] Chevallereau, C., Abba, G., Aoustin, Y., Plestan, F., Westervelt, E., de Wit, C.C., Grizzle, J.: Rabbit: A testbed for advanced control theory. IEEE Control Systems Magazine **23**(5), 57–79 (2003)

[19] Chevallereau, C., Aoustin, Y.: Optimal reference trajectories for walking and running of a biped robot. Robotica **19**(05), 557–569 (2001)

[20] Clarke Jr, E.M., Grumberg, O., Kroening, D., Peled, D., Veith, H.: Model checking. MIT press (2018)

[21] Da, X., Grizzle, J.: Combining Trajectory Optimization, Supervised Machine Learning, and Model Structure for Mitigating the Curse of Dimensionality in the Control of Bipedal Robots. arXiv preprint arXiv:1711.02223 (2017)

[22] Diehl, M., Leineweber, D.B., Schfer, A.A.: MUSCOD-II User's Manual. IWR (2001)

[23] Dynamics, B.: URL `http://www.bostondynamics.com`

[24] Dynamics, B.: Getting some air, Atlas? YouTube (2018). URL `https://www.youtube.com/watch?v=vjSohj-Iclc`

[25] El-Guindy, A., Han, D., Althoff, M.: Estimating the region of attraction via forward reachable sets. In: 2017 American Control Conference (ACC), pp. 1263–1270. IEEE (2017)

[26] Fallon, K.E., Broad, E., Thompson, M.W., Reull, P.A.: Nutritional and fluid intake in a 100-km ultramarathon. International journal of sport nutrition **8**(1), 24–35 (1998)

[27] Gabrielli, G., Von Karman, T.: What price speed?: specific power required for propulsion of vehicles (1950)

[28] Gan, Z., Wiestner, T., Weishaupt, M.A., Waldern, N.M., Remy, C.D.: Passive dynamics explain quadrupedal walking, trotting, and tlting. Journal of Computational and Nonlinear Dynamics **11**(2), 021,008 (2016)

[29] Geyer, H., Seyfarth, A., Blickhan, R.: Compliant leg behaviour explains basic dynamics of walking and running. Proceedings of the Royal Society of London B: Biological Sciences **273**(1603), 2861–2867 (2006)

[30] Goebel, R., Sanfelice, R., Teel, A.: Hybrid dynamical systems. Control Systems Magazine, IEEE **29**(2), 28–93 (2009)

[31] Gong, Y., Hartley, R., Da, X., Hereid, A., Harib, O., Huang, J.K., Grizzle, J.: Feedback Control of a Cassie Bipedal Robot: Walking, Standing, and Riding a Segway. arXiv preprint arXiv:1809.07279 (2018)

[32] Gordon, K.E., Ferris, D.P., Kuo, A.D.: Metabolic and mechanical energy costs of reducing vertical center of mass movement during gait. Archives of physical medicine and rehabilitation **90**(1), 136–144 (2009)

[33] Green, K., Smit-Anseeuw, N., Gleason, R., Remy, C.D.: Design and control of a recovery system for legged robots. In: 2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp. 958–963. IEEE (2016)

[34] Haberland, M., Kim, S.: On extracting design principles from biology: II. Case studythe effect of knee direction on bipedal robot running efficiency. Bioinspiration & biomimetics **10**(1), 016,011 (2015)

[35] Hasaneini, S.J., Macnab, C.J., Bertram, J.E., Leung, H.: Optimal relative timing of stance push-off and swing leg retraction. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3616–3623. IEEE (2013)

[36] Heater, B.: Boston dynamics showcases new uses for spotmini ahead of commercial production (2019). URL `https://techcrunch.com/2019/04/19/boston-dynamics-showcases-new-uses-for-spotmini-ahead-of-commercial-production/`. [Online; posted 19-April-2019]

[37] Henrion, D., Korda, M.: Convex computation of the region of attraction of polynomial control systems. IEEE Transactions on Automatic Control **59**(2), 297–312 (2014)

[38] Hereid, A., Ames, A.D.: FROST: Fast robot optimization and simulation toolkit. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 719–726. IEEE (2017)

[39] Hildebrand, M.: The Adaptive Significance of Tetrapod Gait Selection. Integrative and Comparative Biology **20**(1), 255–267 (1980)

[40] Hoyt, D.F., Taylor, C.R.: Gait and the energetics of locomotion in horses. Nature **292**(5820), 239–240 (1981)

[41] Hsu, S.C., Xu, X., Ames, A.D.: Control barrier function based quadratic programs with application to bipedal robotic walking. In: 2015 American Control Conference (ACC), pp. 4542–4548 (2015). DOI 10.1109/ACC.2015.7172044

[42] Hubicki, C., Abate, A., Clary, P., Rezazadeh, S., Jones, M., Peekema, A., Van Why, J., Domres, R., Wu, A., Martin, W., others: Walking and Running with Passive Compliance. IEEE ROBOTICS AND AUTOMATION MAGAZINE p. 1 (2016)

[43] Hubicki, C., Grimes, J., Jones, M., Renjewski, D., Spröwitz, A., Abate, A., Hurst, J.: Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot. The International Journal of Robotics Research **35**(12), 1497–1521 (2016)

[44] Hutter, M., Remy, C.D., Hoepflinger, M.A., Siegwart, R.: Scarl\emphETH: Design and Control of a PlanarRunning Robot. In: International Conference on Intelligent RObots and Systems, IROS, p. (under review) (2011)

[45] Hutter, M., Remy, C.D., Hoepflinger, M.A., Siegwart, R.: ScarlETH: Design and control of a planar running robot. In: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pp. 562–567. IEEE (2011)

[46] Inman, V.T., Eberhart, H.D., others: The major determinants in normal and pathological gait. J Bone Joint Surg Am **35**(3), 543–558 (1953)

[47] Knox, B.T.: Design of a biped robot capable of dynamic maneuvers. Master's thesis, The Ohio State University (2008)

[48] Kohl, N., Stone, P.: Policy gradient reinforcement learning for fast quadrupedal locomotion. In: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, vol. 3, pp. 2619–2624. IEEE (2004)

[49] Koolen, T., Posa, M., Tedrake, R.: Balance control using center of mass height variation: limitations imposed by unilateral contact. In: Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, pp. 8–15. IEEE (2016)

[50] Korda, M., Henrion, D., Jones, C.N.: Controller design and value function approximation for nonlinear dynamical systems. Automatica **67**, 54–66 (2016)

[51] Kousik, S., Holmes, P., Vasudevan, R.: Safe, aggressive quadrotor flight via reachability-based trajectory design. arXiv preprint arXiv:1904.05728 (2019)

[52] Kousik, S., Vaskov, S., Johnson-Roberson, M., Vasudevan, R.: Safe trajectory synthesis for autonomous driving in unforeseen environments. In: ASME 2017 Dynamic Systems and Control Conference, pp. V001T44A005–V001T44A005. American Society of Mechanical Engineers (2017)

[53] Lasserre, J.B.: Moments, positive polynomials and their applications, vol. 1. World Scientific (2009)

[54] Leineweber, D.: Efficient reduced SQP methods for the optimization of chemical processes described by large sparse DAE models (1999)

[55] Majumdar, A., Ahmadi, A.A., Tedrake, R.: Control design along trajectories with sums of squares programming. In: 2013 IEEE International Conference on Robotics and Automation, pp. 4054–4061. IEEE (2013)

[56] Majumdar, A., Vasudevan, R., Tobenkin, M.M., Tedrake, R.: Convex optimization of nonlinear feedback controllers via occupation measures. The International Journal of Robotics Research p. 0278364914528059 (2014)

[57] Margaria, R.: Sulla fisiologia e specialmente sul consumo energetico della marcia e della corsa a varie velocita ed inclinazioni del terreno (1938)

[58] Matthis, J.S., Fajen, B.R.: Visual control of foot placement when walking over complex terrain. Journal of experimental psychology: human perception and performance **40**(1), 106 (2014)

[59] McGeer, T.: Passive Dynamic Walking. The International Journal of Robotics Research **9**(2), 62–82 (1990)

[60] McMahon, T.A., Valiant, G., Frederick, E.C.: Groucho running. Journal of Applied Physiology **62**(6), 2326–2337 (1987)

[61] Mochon, S., McMahon, T.A.: Ballistic walking: An improved model. Mathematical Biosciences **52**(3), 241–260 (1980)

[62] Mohan, S., Shia, V., Vasudevan, R.: Convex Computation of the Reachable Set for Hybrid Systems with Parametric Uncertainty. arXiv preprint arXiv:1601.01019 (2016)

[63] Motahar, M.S., Veer, S., Poulakakis, I.: Composing limit cycles for motion planning of 3d bipedal walkers. In: 2016 IEEE 55th Conference on Decision and Control (CDC), pp. 6368–6374. IEEE (2016)

[64] Nguyen, Q., Agrawal, A., Da, X., Martin, W.C., Geyer, H., Grizzle, J.W., Sreenath, K.: Dynamic walking on randomly-varying discrete terrain with one-step preview. In: Robotics: Science and Systems (2017)

[65] Nguyen, Q., Hereid, A., Grizzle, J.W., Ames, A.D., Sreenath, K.: 3d dynamic walking on stepping stones with control barrier functions. In: 2016 IEEE 55th Conference on Decision and Control (CDC), pp. 827–834 (2016). DOI 10.1109/CDC.2016.7798370

[66] Nguyen, Q., Sreenath, K.: Optimal Robust Control for Bipedal Robots through Control Lyapunov Function based Quadratic Programs. In: Robotics: Science and Systems (2015)

[67] Nguyen, Q., Sreenath, K.: Safety-critical control for dynamical bipedal walking with precise footstep placement. IFAC-PapersOnLine **48**(27), 147–154 (2015)

[68] Nguyen, Q., Sreenath, K.: Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In: American Control Conference (ACC), 2016, pp. 322–328. IEEE (2016)

[69] Papp, D., Yildiz, S.: Sum-of-squares optimization without semidefinite programming. SIAM Journal on Optimization **29**(1), 822–851 (2019)

[70] Parrilo, P.A.: Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. PhD Thesis, California Institute of Technology (2000)

[71] Posa, M., Koolen, T., Tedrake, R.: Balancing and Step Recovery Capturability via Sums-of-Squares Optimization. In: 2017 Robotics: Science and Systems Conference (2017)

[72] Prajna, S., Jadbabaie, A.: Safety verification of hybrid systems using barrier certificates. In: International Workshop on Hybrid Systems: Computation and Control, pp. 477–492. Springer (2004)

[73] Pratt, J., Carff, J., Drakunov, S., Goswami, A.: Capture Point: A Step toward Humanoid Push Recovery. In: 2006 6th IEEE-RAS International Conference on Humanoid Robots, pp. 200–207 (2006). DOI 10.1109/ICHR.2006.321385

[74] Pratt, J., Chew, C.M., Torres, A., Dilworth, P., Pratt, G.: Virtual Model Control: An Intuitive Approach for Bipedal Locomotion. The International Journal of Robotics Research **20**(2), 129–143 (2001)

[75] Pratt, J., Pratt, G.: Intuitive control of a planar bipedal walking robot. In: Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, vol. 3, pp. 2014–2021. IEEE (1998)

[76] Reher, J., Cousineau, E.A., Hereid, A., Hubicki, C.M., Ames, A.D.: Realizing dynamic and efficient bipedal locomotion on the humanoid robot durus. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1794–1801. IEEE (2016)

[77] Remy, C.D.: Optimal Exploitation of Natural Dynamics in Legged Locomotion. PhD Thesis, Eidgenssische Technische Hochschule (2011)

[78] Remy, C.D., Hutter, M., Hoepflinger, M., Bloesch, M., Gehring, C., Siegwart, R.: Quadrupedal robots with stiff and compliant actuation. at-Automatisierungstechnik Methoden und Anwendungen der Steuerungs-, Regelungs-und Informationstechnik **60**(11), 682–691 (2012)

[79] Robotics, A.: URL `http://www.agilityrobotics.com/`

[80] Robotics, A.: Cassie Sim Comparison. YouTube (2018). URL `https://www.youtube.com/watch?v=kxNeLDB755Q`

[81] Robotics, A.: Cassie: Walk in the park. YouTube (2018). URL `https://www.youtube.com/watch?v=_6XBZHvt7bk`

[82] Ruina, A., Bertram, J.E., Srinivasan, M.: A collisional model of the energetic cost of support work qualitatively explains leg sequencing in walking and galloping, pseudo-elastic leg behavior in running and the walk-to-run transition. Journal of theoretical biology **237**(2), 170–192 (2005)

[83] Saglam, C.O., Byl, K.: Quantifying the trade-offs between stability versus energy use for underactuated biped walking. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2550–2557. IEEE (2014)

[84] Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., Fujimura, K.: The intelligent ASIMO: System overview and integration. In: Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, vol. 3, pp. 2478–2483. IEEE (2002)

[85] Sastry, S.: Nonlinear systems: analysis, stability, and control, vol. 10. Springer Science & Business Media (2013)

[86] Seok, S., Wang, A., Chuah, M.Y., Otten, D., Lang, J., Kim, S.: Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on, pp. 3307–3312. IEEE (2013)

[87] Shia, V., Vasudevan, R., Bajcsy, R., Tedrake, R.: Convex computation of the reachable set for controlled polynomial hybrid systems. In: 53rd IEEE Conference on Decision and Control, pp. 1499–1506 (2014). DOI 10.1109/CDC.2014.7039612

[88] Smit-Anseeuw, N., Gleason, R., Vasudevan, R., Remy, C.D.: The energetic benefit of robotic gait selection: A case study on the robot RAMone. IEEE Robotics and Automation Letters **2**(2), 1124–1131 (2017)

[89] Smit-Anseeuw, N., Gleason, R., Zaytsev, P., Remy, C.D.: RAMone: A planar biped for studying the energetics of gait. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4090–4095 (2017). DOI 10.1109/IROS.2017.8206266

[90] Smit-Anseeuw, N., Vasudevan, R., Remy, C.D.: Walking with confidence: Safety regulation for full order biped models. IEEE Robotics and Automation Letters (2019)

[91] Spong, M.W.: The swing up control problem for the acrobot. IEEE control systems magazine **15**(1), 49–55 (1995)

[92] Sreenath, K., Park, H.W., Poulakakis, I., Grizzle, J.W.: A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on MABEL. The International Journal of Robotics Research **30**(9), 1170–1193 (2011)

[93] Srinivasan, M.: Fifteen observations on the structure of energy-minimizing gaits in many simple biped models. Journal of The Royal Society Interface p. rsif20090544 (2010)

[94] Srinivasan, M., Ruina, A.: Computer optimization of a minimal biped model discovers walking and running. Nature **439**(7072), 72–75 (2006)

[95] Stengle, G.: A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. Mathematische Annalen **207**(2), 87–97 (1974)

[96] Tang, J.Z., Boudali, A.M., Manchester, I.R.: Invariant funnels for underactuated dynamic walking robots: New phase variable and experimental validation. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3497–3504 (2017). DOI 10.1109/ICRA.2017.7989400

[97] Tedrake, R., Zhang, T.W., Fong, M.f., Seung, H.S.: Actuating a simple 3d passive dynamic walker. In: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, vol. 5, pp. 4656–4661. IEEE (2004)

[98] Tucker, V.A.: Energetic cost of locomotion in animals. Comparative Biochemistry and Physiology **34**(4), 841–846 (1970)

[99] Tucker, V.A.: The Energetic Cost of Moving About: Walking and running are extremely inefficient forms of locomotion. Much greater efficiency is achieved by birds, fishand bicyclists. American Scientist **63**(4), 413–419 (1975)

[100] Vaskov, S., Sharma, U., Kousik, S., Johnson-Roberson, M., Vasudevan, R.: Guaranteed safe reachability-based trajectory design for a high-fidelity model of an autonomous passenger vehicle. arXiv preprint arXiv:1902.01786 (2019)

[101] Veer, S., Poulakakis, I.: Safe adaptive switching among dynamical movement primitives: Application to 3d limit-cycle walkers. arXiv preprint arXiv:1810.00527 (2018)

[102] Veer, S., Poulakakis, I., et al.: Input-to-state stability of periodic orbits of systems with impulse effects via poincaré analysis. IEEE Transactions on Automatic Control (2019)

[103] Vincent, J.: Fords vision for package delivery is a robot that folds up into the back of a self-driving car (2019). URL `https://www.theverge.com/2019/5/22/18635439/robot-package-delivery-ford-agility-robotics-autonomous-digit`. [Online; posted 22-May-2019]

[104] Westervelt, E., Grizzle, J., Koditschek, D.: Hybrid zero dynamics of planar biped walkers. IEEE Transactions on Automatic Control **48**(1), 42–56 (2003)

[105] Wieber, P.B.: On the stability of walking systems. In: Proceedings of the international workshop on humanoid and human friendly robotics (2002)

[106] Wieland, P., Allgower, F.: Constructive safety using control barrier functions. IFAC Proceedings Volumes **40**(12), 462–467 (2007)

[107] Xi, W., Remy, C.D.: Optimal gaits and motions for legged robots. In: Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, pp. 3259–3265. IEEE (2014)

[108] Xi, W., Yesilevskiy, Y., Remy, C.D.: Selecting gaits for economical locomotion of legged robots. The International Journal of Robotics Research **35**(9), 1140–1154 (2016). DOI 10.1177/0278364915612572. URL `https://doi.org/10.1177/0278364915612572`

[109] Yang, T., Westervelt, E., Schmiedeler, J.P., Bockbrader, R.: Design and control of a planar bipedal robot ERNIE with parallel knee compliance. Autonomous robots **25**(4), 317–330 (2008)

[110] Zaytsev, P., Hasaneini, S.J., Ruina, A.: Two steps is enough: No need to plan far ahead for walking balance. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 6295–6300. IEEE (2015)

[111] Zhao, P., Mohan, S., Vasudevan, R.: Optimal control for nonlinear hybrid systems via convex relaxations. arXiv preprint arXiv:1702.04310 (2017)