Trajectory Optimization and Machine Learning to Design Feedback Controllers for Bipedal Robots with Provable Stability

by

Xingye Da

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in the University of Michigan
2018

Doctoral Committee:

Professor Jessy W. Grizzle, Chair
Assistant Professor C. David Remy
Assistant Professor Shai Revzen
Assistant Professor Ramanarayan Vasudevan

Xingye Da

xda@umich.edu

ORCID iD: 0000-0003-4439-4980

© Xingye Da 2018

To my family

ACKNOWLEDGEMENTS

The work presented here would not have been possible without the contribution and support of many others. I want to express my most sincere appreciation to Prof. Jessy Grizzle for his continual enthusiasm and unconditional support. I wish to thank the members of my dissertation committee, Prof. C. David Remy, Prof. Shai Revzen, and Prof. Ramanarayan Vasudevan, for their support and insightful feedback. I was privileged to have hardware and software instruction from the members of Agility Robotics, Jonathan Hurst, Damion Shelton, Mikhail Jones, Andy Abate, Ryan Domres, and Daniel Bennett. The code for optimization, simulation, and control originated with Mikhail Jones. Brian Buss and Brent Griffin welcomed me into the lab and patiently trained me to solve real-world problems on MARLO. My experimental work would not have been possible without the collaboration of Omar Harib and Ross Hartley, day and night. I want to thank Yukai Gong for taking care of Cassie Blue and performing outstanding experiments. Ayonga Hereid, Eva Mungai, and Bruce Huang brought a cheerful atmosphere to the lab. To Shelly Feldkamp, Judi Jones, Katherine McAlpine, Catharine June, Nicole Moore, Damen Provost, I owe many thanks for their timely, professional assistance. I extend many thanks to Koushil Sreenath, Quan Nguyen, and Hamed Razavi for our fruitful collaboration. To my fiancée Chengying Luo, I am so grateful to have you in my life. My work has more meaning because I can share it with you. To my family in China and the many friends I met in Ann Arbor, I could not have done this without you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

ABSTRACT

This thesis combines recent advances in trajectory optimization of hybrid dynamical systems with machine learning and geometric control theory to achieve unprecedented performance in bipedal robot locomotion. The work greatly expands the class of robot models for which feedback controllers can be designed with provable stability. The methods are widely applicable beyond bipedal robots, including exoskeletons, and prostheses, and eventually, drones, ADAS, and other highly automated machines.

One main idea of this thesis is to greatly expand the use of multiple trajectories in the design of a stabilizing controller. The computation of many trajectories is now feasible due to new optimization tools. The computations are not fast enough to apply in the real-time, however, so they are not feasible for model predictive control (MPC). The offline "library" approach will encounter the curse of dimensionality for the high-dimensional models common in bipedal robots. To overcome these obstructions, we embed a stable walking motion in an attractive low-dimensional surface of the system's state space. The periodic orbit is now an attractor of the low-dimensional state-variable model but is not attractive in the full-order system. We then use the special structure of mechanical models associated with bipedal robots to embed the low-dimensional model in the original model in such a manner that the desired walking motions are locally exponentially stable.

The ultimate solution in this thesis will generate model-based feedback controllers for bipedal robots, in such a way that the closed-loop system has a large stability basin, exhibits highly agile, dynamic behavior, and can deal with significant pertur-

bations coming from the environment. In the case of bipeds: “model-based” means that the controller will be designed on the basis of the full floating-base dynamic model of the robot, and not a simplified model, such as the LIP (Linear Inverted Pendulum). By “agile and dynamic” is meant that the robot moves at the speed of a normal human or faster while walking off a curb. By “significant perturbation” is meant a human tripping, and while falling, throwing his/her full weight into the back of the robot.

CHAPTER I

Introduction

1.1 Motivation

Bipedal robots have the potential to travel through rough terrain and over human infrastructure for search and rescue missions. This thesis seeks feedback controllers that endow bipedal robots with a large stability basin, exhibit agile, dynamic behavior, and can deal with significant perturbations coming from the environment. Experimental results are shown in Figure 1.1.

Model-based control methods seek to synthesize a feedback controller on the basis of one or more dynamic models of a system. Bipedal robots generally have numerous links connected through joints such as hips, knees, and ankles, which result in a

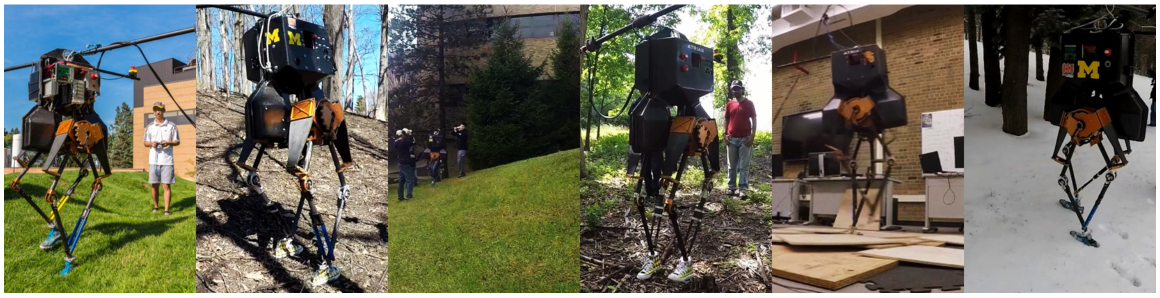


Figure 1.1: MARLO Experiment Summary. MARLO walking blind with passive feet. The feedback algorithms are tested on challenging terrain outdoors. While these images mostly show successes, sometimes the robot fails spectacularly, such as the robot catching fire while the Discovery Channel was filming. Failures, breaking the robot, and repairing it are all important learning experiences.

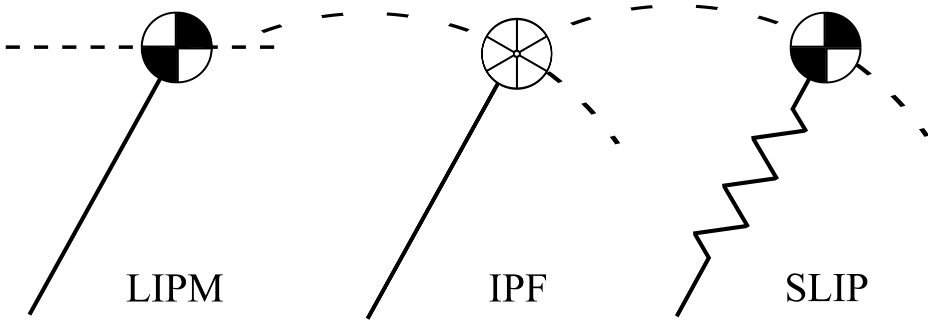


Figure 1.2: Pendulum Models. Adapted from [50, Figure 5]. Three low-dimensional models that are frequently used as approximate representations of walking robots. From left to right: the Linear Inverted Pendulum (LIP) lumps the mass of the robot at a point moving at a constant height and assumes massless legs; the Inverted Pendulum with Flywheel (IPF) relaxes the assumption on constant height and adds a flywheel to account for internal angular momentum; and the Spring-Loaded Inverted Pendulum (SLIP) adds a spring to model a robot's legs as a massless pogo stick. There is no obvious way to embed these low-dimensional models into the full model of a robot.

high degree of freedom mechanical model and therefore in a dynamic model with a relatively high dimensional state space. The most common approach in the literature to getting around the high dimension of the model is to represent a locomotion task through the dynamics of a low-dimensional inverted pendulum (e.g., LIP, SLIP or others shown in Figure 1.2), which when equipped with a foot-placement strategy can exhibit stable walking or running motions [60, 96, 95, 94]. The robot itself is then controlled in such a way that its center of mass approximately follows the target dynamics of the selected pendulum model. The many challenges associated with this more common approach include: achieving stable solutions in the full model; exploiting the full capability of the machine, especially in light of physical constraints of the hardware or environment; deciding how to associate the states of the low-dimensional pendulum with the full-order system; and finally, even deciding upon the appropriate pendulum model for a given task is not evident: what is the correct model for turning while stepping off a platform?

For these reasons, this thesis does not rely on a pre-specified pendulum model to encode a walking motion. Instead, the thesis follows the approach of using the full-order model directly. It seeks to design controllers for high degree-of-freedom (DoF) bipedal robots with several degrees of underactuation (DoU), or, if the robot is "fully actuated", it wishes to take into account the limited ability of ankle torques to affect the overall evolution of the robot. The thesis will focus on the tasks of walking stably forward, backward, or in place, and transitioning among such motions. The gaits will be dynamic in the sense that they can use the full capability of the robot regarding speed, terrain type, and other forms of agility. Moreover, of course, the controller needs to be embedded on the robot for real-time implementation.

The ultimate solution in this thesis is illustrated in Figure 1.3. To overcome the obstructions imposed by high-dimensional bipedal models, the thesis embeds a stable walking motion in an attractive low-dimensional surface of the system's state space. The process begins with trajectory optimization to design an open-loop periodic walking motion of the high-dimensional model and then adding to this solution, a carefully selected set of additional open-loop trajectories of the model that steer toward the nominal motion. A drawback of trajectories is that they provide little information on how to respond to a disturbance. To address this shortcoming, Supervised Machine Learning is used to construct a low-dimensional state-variable realization of the open-loop trajectories. The periodic orbit is now an attractor of the low-dimensional state-variable model but is not attractive in the full-order system. A special structure of mechanical models associated with bipedal robots is then used to embed the low-dimensional model in the original model in such a manner that the desired walking motions are locally exponentially stable.

1.2 Starting Point

The starting point for the ideas developed in the thesis is the Hybrid Zero Dynamics (HZD) method in [126], a successful control approach for directly using the full-order model of a bipedal robot. The thesis first explores a few intuitive approaches

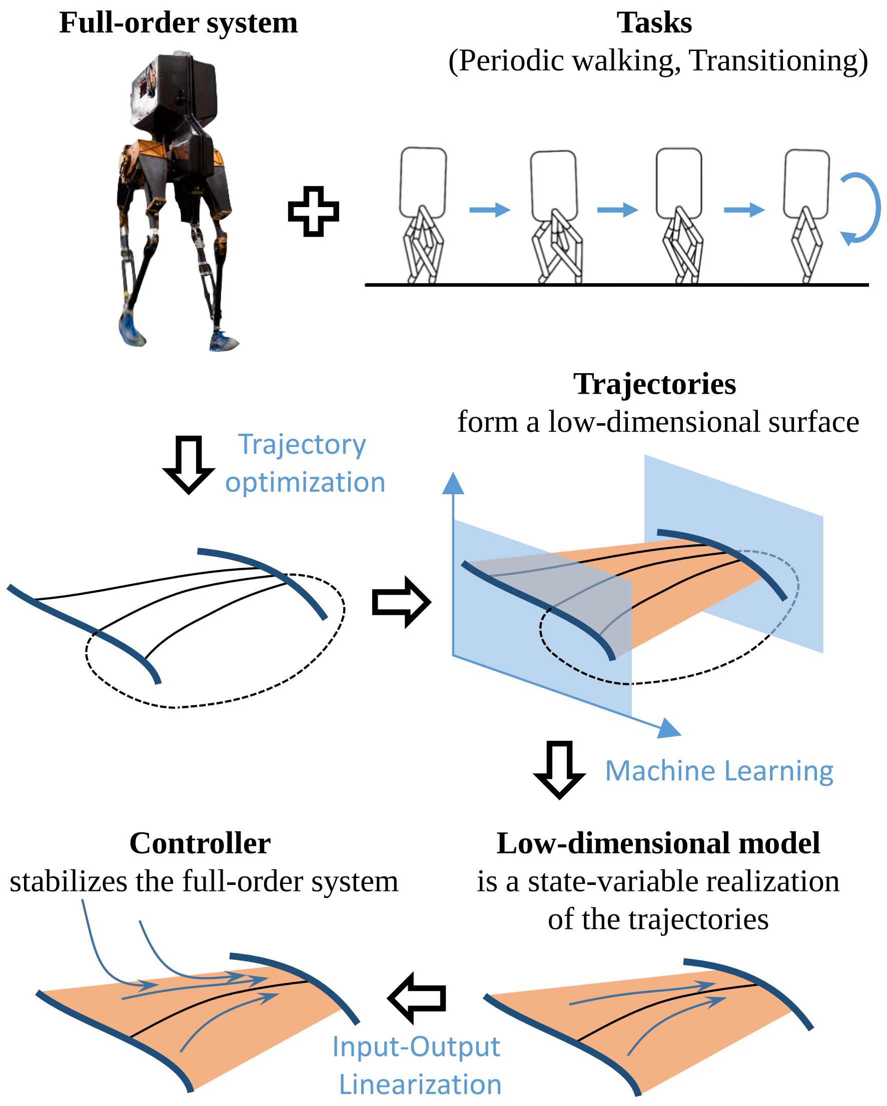


Figure 1.3: The overall approach. The full-order model, the desired objectives, and physical constraints are combined into a trajectory optimization problem for designing a periodic gait. Using model structure or physical insight, a low-dimensional surface of initial conditions is selected for trajectory building, with the trajectories designed to approach the periodic orbit. If the trajectories form a low-dimensional surface, Supervised Machine Learning is used to extract a vector field from the data that realizes the trajectories. System structure is then used again to render the low-dimensional model (surface and vector field) invariant and attractive.

to overcome numerous drawbacks of both basic 2D walking results and state-of-the-art 3D walking results. A formal approach is then given in subsequent chapters with new theorems for control design with stability proofs.

1.2.1 Basic HZD: Strengths and Weaknesses

In HZD, stable walking is identified with asymptotically stable periodic orbits in a hybrid dynamic model. The periodic orbit is generated through trajectory optimization of the full-order hybrid model. If the model satisfies the following conditions:

- There exists a hybrid invariant surface in the state space.
- The periodic orbit is asymptotically stable in the surface.
- The transverse dynamics are feedback linearizable.

Then a state-variable feedback can be found to asymptotically stabilize the orbit in the full-order model.

In the case of planar robots with only one degree of underactuation, these conditions turn out to be easy to meet. While this theory has been successfully implemented on many robots [7, 5, 24, 55, 74, 116, 115, 101], lower-limb prostheses [45, 2, 132] and even an exoskeleton for hands-free walking [3], it has important limitations that this thesis overcomes:

- In basic HZD, only one optimization is done, namely the determination of the periodic orbit. Hence, only that solution is guaranteed to be feasible (i.e., in addition to satisfying the mechanical model, feasible solutions must respect actuator limitations and ground reaction force specifications, for example). Here, an entire (low-dimensional) surface of feasible solutions is built.
- For robots with one degree of underactuation, the stability mechanism in [125, Ch. 5.4] relies on energy loss at impacts, similar to the stability proofs for

passive robots walking down a slope. Here, more general stability mechanisms in bipedal robots [94], such as posture adjustments through foot placement and knee bend, automatically arise.

- Only gaits for which a monotonic variable can be identified can be treated with the basic HZD method. Here, a much richer set of locomotion primitives can be realized, such as stepping in place or transitioning from walking forward to walking backward. As in [120, 100, 32], the feedback is allowed to be time-dependent, enriching the set of possible closed-loop solutions.

Importantly, it has proven difficult to meet the basic HZD conditions for 3D robots, in general, or even for planar robots with more than one degree of underactuation. In the few cases where researchers have achieved asymptotically stable 3D walking, the closed-loop performance has shown a limited ability to recover from environmental uncertainty or force perturbations. In this light, two main approaches are discussed next and compared to the methods in this thesis.

1.2.2 Bilinear Matrix Inequalities for Local Exponential Stability

The method proposed in [20, 51], based upon the (Jacobian) linearization of the Poincaré map, seeks to achieve local exponential stability, aka orbital stability, for robots with more than one degree of underactuation. For such robots, a periodic orbit developed through HZD has no *a priori* guarantee of asymptotic stability within a hybrid invariant surface designed through optimization. The method of [20, 51] begins by replacing the nominal invariant surface by a parameterized family of surfaces, where each surface has the same dimension, contains the given periodic orbit, and one of the surfaces is the original surface. A bilinear matrix inequality (BMI) optimization problem is then set up to search for parameters so that the Jacobian of the Poincaré map has its eigenvalues in the unit circle. In other words, it seeks to reshape the

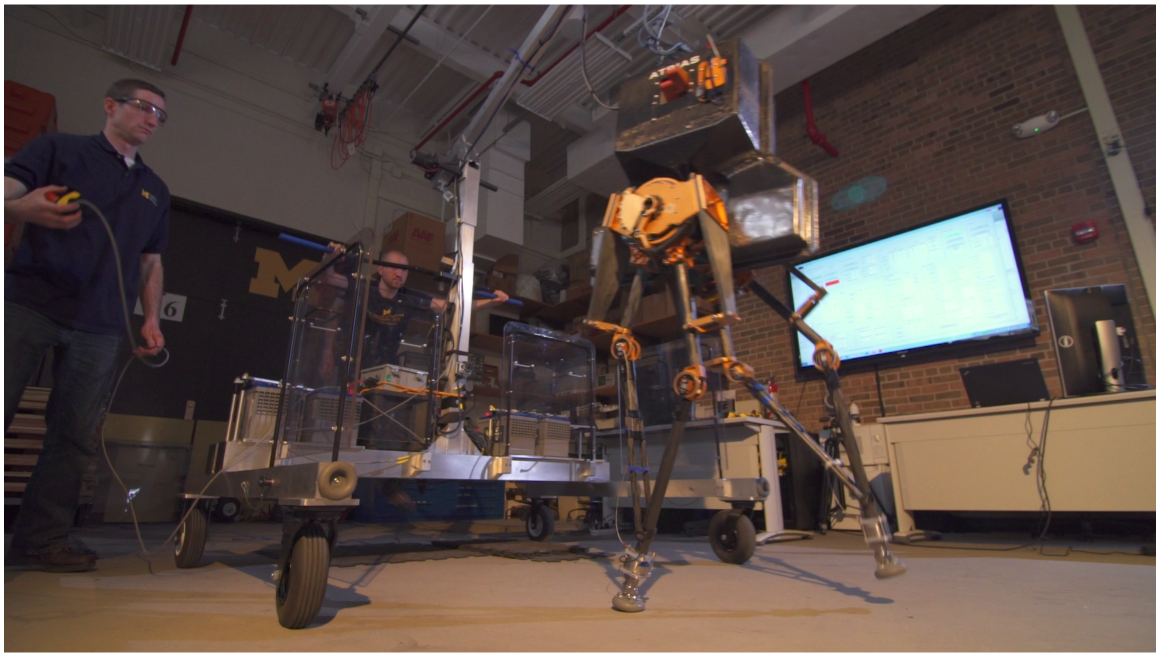


Figure 1.4: MARLO walks in the laboratory over 20 steps on point feet.

invariant surface so that the given periodic orbit is locally exponentially stable in the surface.

The parameter search may or may not be successful, and designs based on Jacobian linearization may not result in an adequate region of attraction. In practice, the BMI-based controller gave the bipedal robot MARLO the ability to consistently traverse the length of the laboratory, a distance of 20 to 25 steps. Impressively, this was accomplished on point feet, which as far as we know, had never been done before. Testing on rough terrain or under impulse-like shoves was not done. Moreover, choosing an "effective" parameterization for the invariant surface can be hard to do.

1.2.3 Robust Optimization of Orbits

The approach proposed in [48] avoids linearization in the controller design and allows the initial periodic orbit to change so as to enhance stability robustness. It also can handle more than one degree of underactuation.

One way to think about its underlying premise is to encode a numerical version of

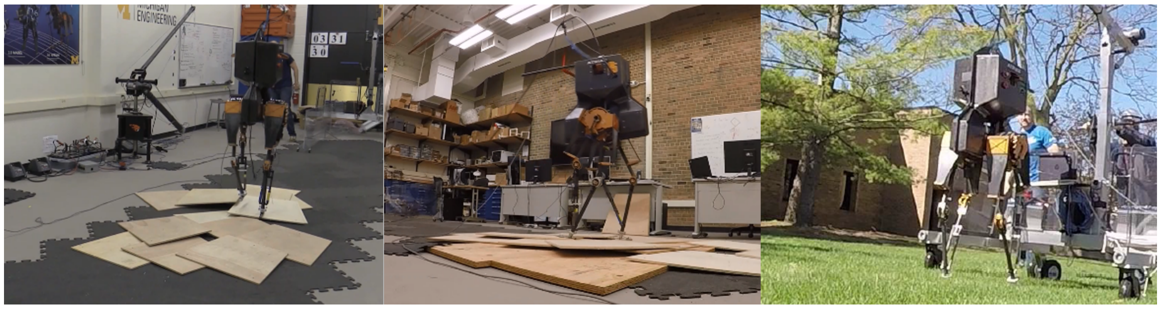


Figure 1.5: MARLO walks on boards and on the grass.

Sontag's famous Input-to-State Stability [112], where the "inputs" are disturbances applied to the model. The key idea is to set up an optimization problem that designs both a periodic orbit and a robust controller simultaneously. The optimization problems have the usual term to assess energy consumed per distance traveled, subject to constraints on periodicity, walking speed, foot clearance, actuator limitations, ground reaction forces, etc. In addition, there are additional terms in the cost function that measure the ability of the system to converge to the periodic orbit after perturbation. The perturbations considered include ground height changes and deviations in center of mass velocity, which are not necessarily small (i.e., local). The optimization problem so posed allows the periodic orbit to be reshaped in order to minimize the cost function. Thus, it tends to find orbits that are "easier to stabilize" in the hybrid invariant surface.

The controller was validated experimentally on MARLO, allowing the robot to walk over structured piles of 5cm boards as well as boards randomly thrown across the laboratory floor. In addition, the robot walked on a variety of surfaces outdoors, including on grass at a speed of 0.9 m/s to 0.95 m/s as shown in Figure 1.5. The closed-loop robustness has been significantly improved compared to the method in [20, 51], but is still not good enough for large terrain variations or force perturbations.

Similar to the method proposed in [20, 51], the controller parameterization (and hence, its structure) must be specified before beginning the optimization. Each perturbation requires simulating the dynamics for multiple walking steps which leads to a large parameter optimization problem. The method may have difficulty scaling up to more diverse perturbations. Moreover, the approach does not come with a formal proof of closed-loop stability.

The work in [48] has, however, two fundamental ideas that this thesis builds upon:

- it uses trajectories other than the periodic orbit itself in the design of the feedback controller; and
- the feedback controller breaks away from the now-classical paradigm of "holonomic virtual constraints" introduced in [126], that is, relative degree two output functions parameterized by a strictly increasing gait timing variable.

1.3 Main Ideas of This Dissertation

One main idea of this dissertation is to greatly expand the use of multiple trajectories in the design of a stabilizing controller. The computation of many trajectories is now feasible due to new optimization tools. The optimization methods used in [126, 48] are based on single shooting. The highly nonlinear, non-convex parameter optimization problem based on [126] takes approximately one hour on a current laptop to find a feasible periodic orbit for a nine degree of freedom, 3D biped model with two degrees of underactuation. The computational time can be much worse if one does not have a good initialization. For the robust optimization problem of [48], the solution time grows to more than four hours for the same robot.

Very importantly, the new optimization tool introduced in [58, 53] returns solutions much faster and with far fewer local optima. On the same laptop used above, it takes about 2 minutes to solve the optimization problem for the above 3D model *with a random initial guess.* Empirical results in [58, 53] show that when seeded with randomly generated initial guesses, 79 of 100 trials converged to the same solution. For a simpler 2D model, it takes about 2 seconds to find a solution, and almost any random guess converged to the same solution. This new and powerful tool allows the exploration of many different combinations of cost function and constraints in a short amount of time, and, of course, the computation of hundreds of trajectories in the time it used to take to find only one.

An intuitive manner to employ multiple trajectories is to build a controller around an entire family of periodic orbits, instead of a single periodic orbit. For example, a set of orbits, O, is designed where each orbit corresponds to a unique forward walking speed v. Assume for each $v_{\min} \leq v \leq v_{\max}$ there exists a state-feedback controller

$$u = \Gamma_v(x),$$

where $x \in \mathbb{R}^m$ is the robot's state vector, that renders the associated orbit O_v locally exponentially stable. Under the assumption that the commanded walking speed, v, "varies sufficiently slowly," the robot's speed can be increased and decreased while maintaining stability. In this way, speed transitions are achievable. The stability of each orbit relies on the controller Γ_v and the stability of the transitions relies on v changing slowly. In the control community, this methods goes under the name of Gain Scheduling.

Building on the method of gain scheduling, this thesis proposes a novel policy for transitioning among orbits that, empirically, in any case, *achieves a larger stability region.* Suppose there exists a map

$$\pi : \mathbb{R}^m \rightarrow [v_{\min}, v_{\max}],$$

projecting the state to forward speed, such that, for $x \in O_v$, $\pi(x) = v$. Consider then a controller given by

$$u = \Gamma_{\pi(x)}(x).$$

Now, when the robot's state is perturbed away from a given walking speed, the control policy attempts to remap itself to one that is "near the new walking speed". The intuition for this policy resulting in a larger basin of attraction is that, whereas for a given speed v, the controller Γ_v respects key physical constraints, when the robot undergoes a larger perturbation, it typically fails (e.g., the robot falls) because Γ_v solicits actions that are beyond the robot's capabilities (or the capabilities of the surface it is walking on). If the perturbed state is near a new periodic orbit for a different walking speed, say $\bar{v}$, then applying the controller $\Gamma_{\bar{v}}$ is more likely to respect the constraints and keep the robot upright.

An interesting analytical phenomenon is associated with the above control policy. Suppose that the robot is walking in steady state at speed v, and in one step, a perturbation moves it exactly to a new steady-state speed, $\bar{v}$. Suppose further that the disturbance now disappears. Then under the control policy $u = \Gamma_{\pi(x)}(x)$, the robot will continue walking at the new speed, $\bar{v}$. The closed-loop control system is hence "neutrally stable": the eigenvalues of the Poincaré map have a magnitude less than one except for the eigenvalue associated with the forward speed, which is exactly one. In other words, the closed-loop robot is acting like an integrator. An additional proportional controller with hand-tuned gain which provides little control effort will render the system locally exponentially stable. To extend the idea further, the thesis also discusses a map to both walking speed and changes in terrain height, where the robot will recenter its orbit to one associated with current speed and ground slope.

As an aside, it is noted that when doing the above work, the dissertation also addresses stepping in place, $v = 0$, as well as walking backwards, $v < 0$. Consequently, gait design based on a monotonically increasing gait timing variable had to be abandoned in favor of a more general control policy based on time; see also [120, 100]. While this additional generality is quite important in its own right, exploring it now would interrupt the flow of ideas. The discussion is therefore delayed to later in the dissertation.

The next chapter in the story of multiple trajectories is to include transient trajectories instead of only periodic trajectories. The motivation is to design a controller that is more robust, with a faster settling time (i.e., no longer constrained to v changing slowly, as in gain scheduling), and also free of the hand-tuned component. The trajectory optimization can be set up as a finite-time convergence problem over a few steps. If the optimization were solvable in real-time, a model predictive control (MPC) method would result. However, this would requirer to solve the optimization at least a thousand times faster than current methods. Doing explicit MPC, that is, computing control solutions offline and parameterizing them in some manner for online use, encounters the curse of dimensionality. For a high-degree-of-freedom robot, the required number of sampled solutions can easily exceed a trillion.

A naive solution is to randomly generate a few trajectories and use regression or supervised learning to fit a mapping from state to action. The hope is to learn a general feedback policy from sparse samples of control actions. This approach works occasionally, but a more theoretically sound method is needed.

The main result of this thesis is a formal approach for using transient trajectories in controller design. A carefully selected surface of initial conditions is used to generate transient trajectories and also a periodic orbit. The transient trajectories are constructed to "fill out" an invariant surface in the state-time space in which the periodic orbit is asymptotically stable. The surface also satisfies the assumptions for HZD, and thus, a feedback can be designed to stabilize the orbit in the full-dimensional state space. The overall method is summarized in Figure 1.3.

1.4 Contributions

The work presented in this thesis contributes to the well-established framework of HZD and greatly enhances it using Machine Learning tools.

1.4.1 Contribution to HZD

The weakness of the previous work on HZD was that it struggled to handle robots with more than one degree of underactuation, which is very common in 3D robots. Moreover, even for a planar robot, the theory presented in [126] for the HZD method assumed a monotonically increasing phase variable that did not exist for stepping-in-place gaits, where the speed is zero. In the first part of the thesis, the HZD work is generalized to 3D walking without using a phase variable. Hence, it allows the robot to walk forward, step in place, and walk backward. The work further combines a set of periodic motions (including walking on slopes) to design a uniform controller that both enlarges the region of attraction for each individual gait and allows smooth transitions between periodic motions.

This thesis later designs motions that exploit the robot's full capabilities while respecting actuator limitations, ground contact forces, and terrain variability. To overcome the limitation of MPC and the curse of dimensionality, the approach in the thesis only samples on a thin set of the state space while it still asymptotically stabilizes the desired walking motion in the full-order model of the robot.

1.4.2 Contribution to Machine Learning

Imitation learning, or teaching from demonstrations, requires good teaching examples to train the controller. The examples are in general time-costly to obtain. For a high-dimensional system, the number of required examples can easily reach a trillion because of exponential dependence on dimension. If a controller can be obtained by imitation learning, there is still no proof of closed-loop stability.

This thesis only uses a small number of training examples to design a vector field in a low dimensional surface on which the stability of a periodic orbit is guaranteed. To stabilize the states that are "not on the surface", a special structural feature of mechanical models is invoked that allows HZD theory to be applied for the design of a feedback that allows the periodic orbit to be asymptotically stable in the full state space.

1.4.3 Broader Impacts

This thesis focuses on developing control design tools that have been successfully implemented on the robots MARLO and Cassie in outdoor experiments. They have also been applied to an exoskeleton that achieves hands-free dynamic walking [52] and to an autonomous truck that adapts to road curvature and side winds [29]. Both projects use the theoretical constructs of this thesis for their controller design. These generalizations demonstrate that a broader class of nonlinear dynamical systems can benefit from the work in this thesis.

1.5 Overview of Thesis

The remainder of this thesis is structured as follows: Chapter II reviews the literature related to the thesis. Chapter III provides a description of the robot that will be used in experiments and introduces two models for it: a 3D floating-based model and a planar model. Chapter IV presents our first attempt to use multiple trajectories (gait library) in the design of a controller for walking. Experimental work is presented that uses the planar model to design a controller that is implementable on a 3D robot. Chapter V uses a naive machine learning method to train a control policy. It presents a gait library with richer features. augmenting periodic gaits for speeds and terrain height with transient gaits. The formal use of machine learning is demonstrated in Chapter VI. Mathematical theory and experimental work are provided. Finally, Chapter VII summarizes the conclusions and discusses future directions.

CHAPTER II

Literature Overview

2.1 Reduced Order Model

Pendulum models are a ubiquitous means in the bipedal robotics literature to reduce the computational burden of full-order models. The linear inverted pendulum (LIP) model is especially prevalent for the design of flat-footed walking gaits based on the Zero Moment Point criterion [59, 130, 84, 106, 61, 88, 90]. Numerical optimization or closed-form computation is used to provide CoM trajectories and swing foot positions for a reduced-order model. A low-level controller and inverse kinematics then realize these on the full-order model or robot. Recent experimental uses of this approach can be found in [93, 68, 41, 104].

The bottom line, however, is that with existing methods, when a pendulum model is pre-specified as a template [42], the full-order model needs to compromise its achievable motions to follow the template. Moreover, for each different task of the robot, such as walking or running, the designer is faced with the selection of the "best" target model. In our approach, a low-dimensional model is generated from the full-order model and the task. It is dynamically feasible and uses the full capability of the robot to accomplish the task.

2.2 Online Optimization

One of the earliest applications of online optimization in bipedal walking was done on a 5-degree-of-freedom simulation model of RABBIT [13, 14]; the computation time for each sampling period was 37.08 s. More recently, Model Predictive Control (MPC) was applied in the DARPA Virtual Robotics Challenge [40]. In that work, the computation time of the MPC solver was important, and a "real-time implementation" on a full-order dynamic model of Atlas was achieved through the use of a novel physics engine and a relaxed contact map. Experimental results on a humanoid robot HRP-2 were reported in [64]. The robot did not walk, but could balance while standing and track a ball with its hands. MPC was applied to the full kinematics and centroidal dynamics of Atlas in [69], and resulted in walking at 0.4 m/s. On a planar biped, higher walking speeds from 0.43 m/s to 0.97 m/s are achieved in [54] using online Hybrid Zero Dynamics (HZD) gait generation. The online optimization generates a new controller based on the commanded speed and updates it at the beginning of the next step. Average computational time is 0.4964 s.

Online computational burden has been reduced by using reduced-order models to compute CoM trajectories and swing foot positions. A low-level controller and inverse kinematics then realize these on the full-order model or robot. Recent experimental uses of this approach can be found in [93, 68, 41]. Though a reduced-order model may provide fundamental insight into the dynamics of a robot [42], with existing implementations, it limits the achievable motions of the robot, and different tasks, such as walking and running, typically require different models.

2.3 Gait Library

A means to get around the limitation of online computation is to pre-compute a set of controllers and design a control policy to "stitch" them together. A policy that switches the task (target walking speed, running vs. walking, stairs vs. flat ground) is employed in [117, 74, 91]. Finite-state machines or motion primitives are used in [86, 105, 73] for rough terrain and in [10] for reducing settling time. Interpolation among gaits has been used to create a continuous family of gaits in [39, 32, 82, 83]. Transient trajectories that approach the nominal periodic orbit were added in [72, 33] to enlarge the basin of attraction. This thesis provides a formal mathematical framework for the work in [33] and increases its applicability.

2.4 Supervised Learning and Reinforcement Learning

Imitation learning is a means to implement supervised learning for control. It uses example trajectories from human motion or numerical optimization with the goal of generalizing them to a controller that can imitate the trajectories. If these trajectories and their interpolation cover the full state space, as in Nvidia's end-to-end self-driving car [17], then the system is stable. If the system is linearizable along the trajectory, such as the Stanford helicopter [81, 1], iterative LQR will locally stabilize the system, though model uncertainty has to be carefully calibrated. For a high dimensional hybrid system, neither of these conditions is satisfied. To reduce the number of required trajectories, reinforcement learning provides a gradient descent method (policy gradient or Q-learning) to sample the trajectories in a particular direction. The method has been verified on a Darwin robot for walking and turning in [79]. To achieve walking experimentally and to mitigate the effects of model uncertainty, the nominal trajectory associates a set of neighboring trajectories with various model parameters. In this thesis, instead of training with neighbors, the trajectories come from a wide range of parameters (e.g., speed from -0.8m/s to 0.8m/s). This wide range potentially defines a large region of attraction. Other recent reinforcement learning [71, 107, 35] methods claim to find control policies for legged robots, but they have yet to demonstrate the ability to deal with model uncertainty, signal noise, and actuator limitations.

2.5 Hybrid Zero Dynamics

A general discussion of Hybrid Zero Dynamics (HZD) has been presented in Chapter I. In this section, we will highlight one particular aspect of HZD, namely, virtual constraints, so that they can be used in subsequent chapters.

In [126], a monotonically increasing function of a robot's generalized coordinates q is first identified, often denoted by $\theta(q)$, and then a family of virtual constraints of the form

$$y = h_0(q) - h_d(\theta(q), \alpha) \tag{2.1}$$

are posited, where $\dim(y) = \dim(u)$, the number of inputs, $h_0(q)$ represents quantities to be regulated, such as knee angles and hip angles, and $h_d(\theta(q), \alpha)$ is a vector of splines representing the to be determined desired evolution of $h_0(q)$. A parameter optimization problem is posed to select the values of α (if they exist) so that $y \equiv 0$ along a periodic solution of the model, torque bounds are met, as are ground contact forces. If the outputs y have vector relative degree two [125, pp. 119], the robot model with outputs (2.1) is input-output linearizable, and hence a feedback controller can be designed that drives the virtual constraints asymptotically to zero. If the surface defined by the outputs being zero is invariant in the hybrid model of the robot, then [125, Ch. 5] provides strong stability theorems for the closed-loop system.

CHAPTER III

Robot Model

3.1 Robot Description

The bipedal robot shown in Figure 3.1 is capable of 3D walking. Its total mass is 63 kg, with approximately 50% of the mass in the hips which house the four motor-harmonic drive assemblies for leg motion in the sagittal plane, and 40% of the mass in the torso, which houses the motors for the lateral motion of the legs and all of the electronics. The legs are very light and are formed by a four-bar linkage. The robot is approximately left-right symmetric. A more complete description is available in [97].

The configuration variables for the system can be defined as

$$q := (q_z, q_y, q_x, q_{1R}, q_{2R}, q_{3R}, q_{1L}, q_{2L}, q_{3L}) \in \mathbb{R}^9.$$

The variables (q_z, q_y, q_x) correspond to the world frame rotation angles: yaw, roll, and pitch. On the other hand, the variables $(q_{1R}, q_{2R}, q_{3R}, q_{1L}, q_{2L}, q_{3L})$ refer to local coordinates, shown in Fig. 3.2. These second set of coordinates are also actuated, resulting in 6 degrees of actuation $u \in \mathbb{R}^6$ and 3 degrees of underactuation. It should be noted that the springs are assumed to be sufficiently stiff and have been deliberately neglected from the model. Also note that for control purposes, the leg coordinates (q_1, q_2) are rewritten as leg angle and knee angle (q_{LA}, q_{KA}), where $q_{LA} := \frac{1}{2}(q_1 + q_2)$

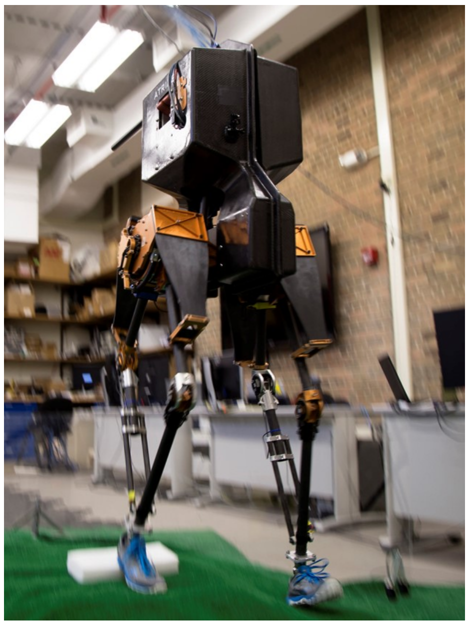


Figure 3.1: MARLO, an ATRIAS 2.1 bipedal robot. ATRIAS-series robots were designed by Jonathan Hurst and the Dynamic Robotics Laboratory at Oregon State University. (Photo: Evan Dougherty)

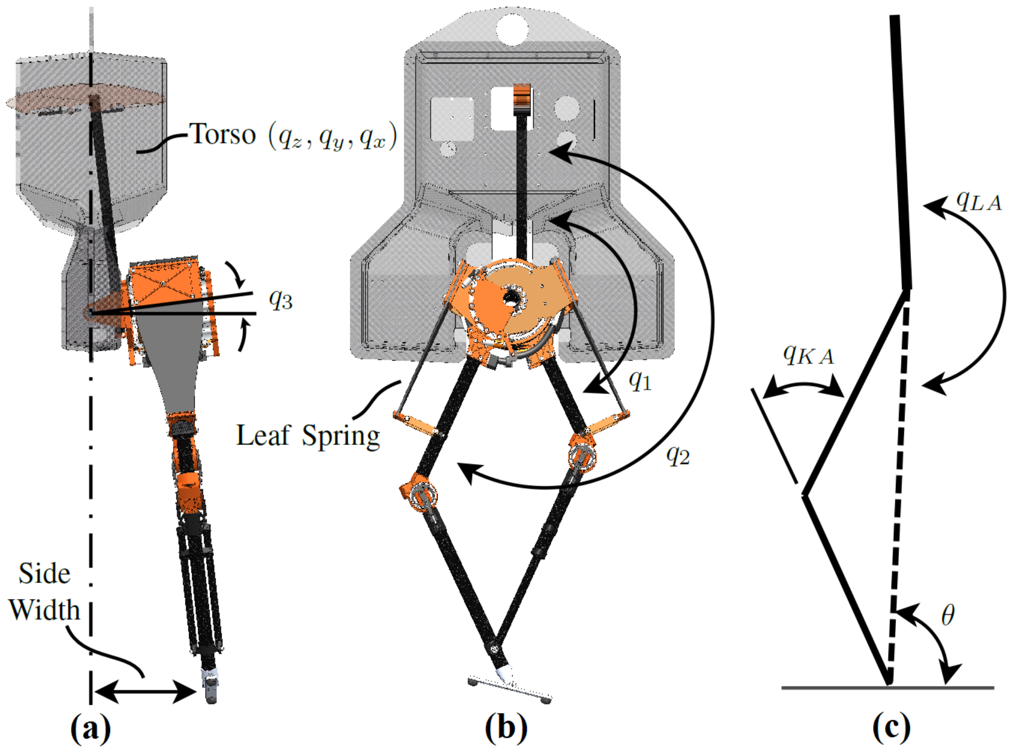


Figure 3.2: Biped coordinates. (a) Lateral plane. (b) Sagittal plane. (c) Equivalent sagittal model.

and $q_{KA} := q_2 - q_1$.

The actuators on the legs that drive the coordinates (q_1, q_2) each operate behind a 50:1 harmonic drive. For those motors, the power amplifiers currently on the robot allow the motors to generate torques up to 5 Nm as opposed to a maximum of 3 Nm, as was the case in previous work [21].

The models used for controller design, analysis, and simulation were presented in [97]. In particular, the 2D representation is obtained from the 3D model by constraining $(q_y, q_z, q_{3L}, q_{3R})$ to zero; see [97, Sec. 4.5].

3.2 Yaw Reduction via Foot Design

The robot has been previously operated with a selection of different feet designs [22, 21]. In this thesis, the two-point-contact passive foot design of [38], shown in Figure 3.3, has been adopted. The foot is composed of a revolute ankle that is connected to the leg (see Figure 3.2(b)) and an arc that bridges two rubber pads. The ankle is free to rotate along a shaft on the arc, freeing the robot's pitch motion. In addition, the narrowness of the foot allows the robot to freely roll as well.

As opposed to the “hoof” design used in [21], the 20 cm distance between the rubber contacts in the current foot counters yaw motion. Furthermore, in contrast to the prosthetic foot used in [22], which could initiate contact at the front or the back of the foot, the current foot acts like a point contact.

Ideally, with this foot in single support, the robot has two degrees of underactuation corresponding to pitch and roll. In practice, yaw is significantly reduced but not eliminated, as discussed in Section 4.6.

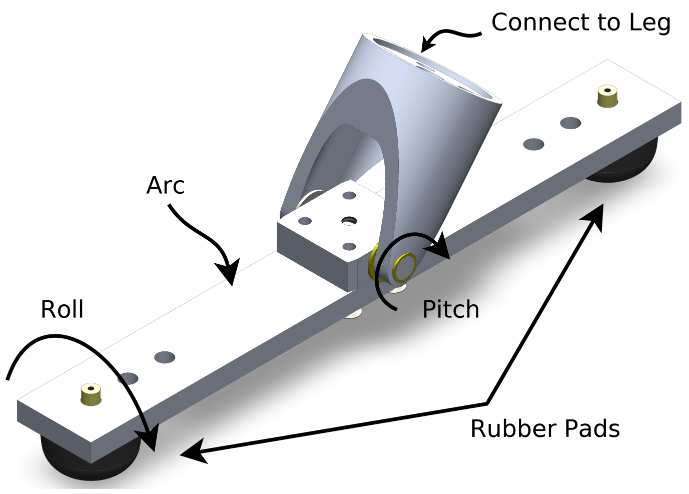


Figure 3.3: A two-point-contact foot was introduced in [38]. Polyurethane rubber pads on the toe and heel reduce yaw, while leaving roll and pitch about the leg end free. The same foot is used in this thesis.

CHAPTER IV

Gait Library of Periodic Orbits

This chapter presents a gait library method to design a feedback controller. As an aside, the work also takes a walking gait that has been designed and stabilized on the basis of an underactuated planar or 2D bipedal model, and shows how to implement it on the 3D underactuated bipedal robot MARLO. The lateral and sagittal planes of the robot are feedback controlled in a decoupled manner, while rotation in the transverse plane, hereafter referred to as yaw, is mechanically limited through narrow passive feet of non-zero length [38].

A large portion of the bipedal robotics literature considers planar models because they are simpler to understand, easier to analyze, and faster to simulate and optimize, to name just a few reasons. The authors of these papers, members of this thesis's lab included, typically argue, imply or hope that insight or progress made on analysis and control methods for planar models serves as a valuable stepping stone to successful deployment on physical robots in 3D. The present chapter will both support this line of thinking with analysis and experiments for at least one instance of a control design method, and offer words of caution where limitations of this approach are perceived.

4.1 Background

Research on passive bipedal walkers began with planar models [43, 77, 78]. It has been taken into the 3D realm in at least two ways. Kuo included roll dynamics with lateral hip actuation on an otherwise passive walker [70]. He showed that the open-loop system did not have stable limit cycles and then proposed a linear state variable feedback for the lateral dynamics to recover asymptotic stability when walking down shallow slopes. Collins et al. stabilized a truly passive walker in 3D through clever mechanical design of the feet [27], which coupled roll and yaw motions of the robot, though the domain of attraction of the limit cycle was impractically small. The Cornell Ranger walked unsupported in 3D for 65 km [16]. Though the robot was powered, it used very minimal actuation in the sagittal plane and achieved passive (or mechanical) stabilization in the lateral plane through specially designed legs. A number of Wisse's robots have used passive means to achieve adequate lateral stabilization so that an essentially sagittal-plane robot could move about unsupported in 3D [26, 127, 128].

Moving to the other end of the actuation spectrum, fully actuated bipedal robots, Ames et al. developed a rigorous geometric framework to decouple the lateral and sagittal plane dynamics through functional Routhian Reduction[1] [9, 110, 111]. With this method, a controller derived from an appropriate planar model can provably create and stabilize a limit cycle in a 3D walker with yaw constrained to zero. The result was experimentally confirmed on NAO [8, 91], which achieved dynamically-stable forward walking at 15 cm/s. Gregg showed analysis and simulations of Routhian Reduction when yaw was un-actuated and viscous friction at the foot provided "rotational stabilization" [44, 46]. Kajita et al. first developed the linear inverted pendulum (LIP) model in 2D and demonstrated experimentally its application to bipedal walk-

[1]This is related to reduction through conserved quantities, where, roughly speaking, the time rate of change of the momentum map evolves as a function of a cyclic variable.

ing [60]. The 2D and 3D LIP models proved fundamental in Pratt et al.'s work on capture point [67, 93]. Hosoda appears to implement decentralized sagittal and lateral control strategies on the pneumatically powered 3D biped Pneuman [56], though the control algorithm is only partially described.

In the middle of the actuation spectrum, hoppers were early examples of underactuated legged robots [96, 95]. Raibert developed the fundamental control strategy for Spring-Loaded Inverted Pendulum (SLIP) models and demonstrated that decoupled and identical sagittal and lateral plane controllers were adequate for control of his hoppers in 3D [96, 95]. Raibert's original SLIP-based strategy, perhaps augmented with foot placement [94, 92], has been extended to the family of robots created by Boston Dynamics [19]. Hurst designed the ATRIAS 2.1 series[2] of bipedal robots to instantiate approximately a 3D SLIP model [102]. The robot's legs are formed through a four-bar mechanism and series compliance actuation is implemented with leaf springs. The feet, when point-feet are not being used, are passive. Inspired by Raibert's SLIP-based controllers, Rezazadeh et al. designed decoupled lateral and sagittal plane controllers for ATRIAS [104], and demonstrated the robot's agility at the 2015 DRC [37]. The yaw motion of the robot was regulated passively through the feet shown in Fig. 3.3, which allow the leg end to act as a pivot with respect to pitch and roll, while yaw is limited through rubber pads attached to a narrow forward pointing bar that is approximately 20 cm in length.

4.2 Contributions and Organization of This Chapter

This chapter focuses on the process of designing a feedback controller on the basis of a planar bipedal model, and achieving a stable walking gait, both indoors and outdoors, on a 3D underactuated bipedal robot. Decoupled lateral and sagittal plane controllers will be used. The controller design builds upon previous work in [104, 21].

[2]ATRIAS stands for Assume the Robot Is A Sphere, emphasizing a 3D pendulum model.

The primary contributions of the present chapter include:

- An orbit updating method is developed to continuously transition over a wide range of walking speeds. Relevant recent references are [123, 21], which designed gaits for a fixed forward speed.
- The sagittal controller design provides systematic and generalizable methods that consider the dynamic model and respect physical constraints commonly found in legged locomotion. The controller in [104, 37] was based on a SLIP model and hand tuning of parameters, which is limited to a specific class of robots.
- A means of unifying time- and phase-based controllers is presented.

The remainder of the chapter is organized as follows. A planar model of the robot is used in Section 4.3 to design a family of stabilized gaits, at speeds varying from stepping in place to walking at 0.8 meter per second. A means to track a commanded speed profile is introduced. A foot-placement controller to stabilize the lateral dynamics during straight-line walking is given in Section 4.4. Section 4.5 provides model-based analysis and simulation of the lateral and sagittal plane controllers on a 3D model of the robot. Section 4.6 reports on experiments with the controller, both indoor and outdoors, while Sect. 4.7 discusses advantages and disadvantages of the approach taken in the chapter.

4.3 2D Gait Design and Stabilization

This section uses the 2D model to design an overall feedback controller that can walk in place, transition to, and maintain a desired walking speed, while employing walking gaits that are suitable for the conditions of the current step and respect mechanical constraints of the robot and environment. The basic walking controller is

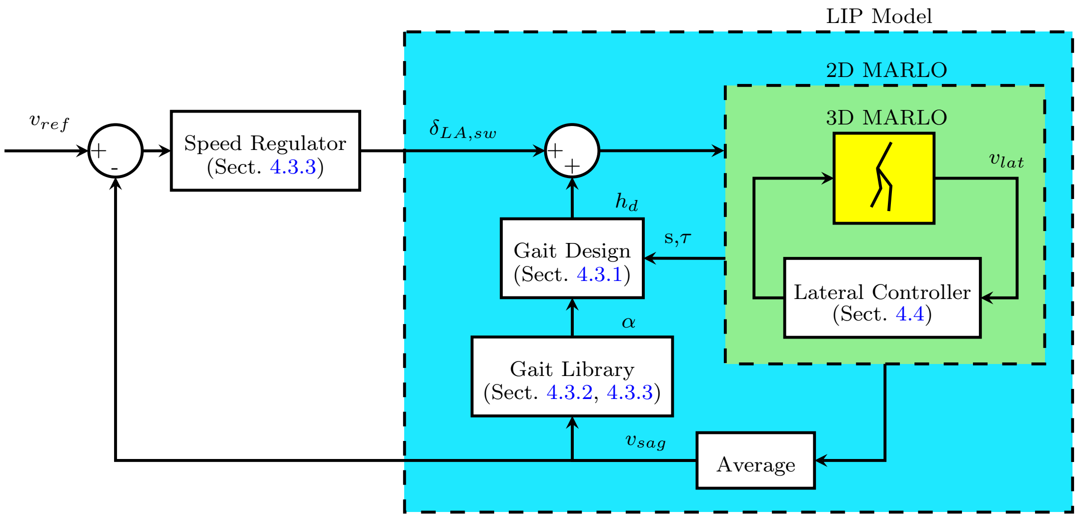

LIP Model
2D MARLO
3D MARLO
v_{ref}
+
-
Speed Regulator (Sect. 4.3.3)
$\delta_{LA,sw}$
+
+
v_{lat}
h_d
s,τ
Gait Design (Sect. 4.3.1)
Lateral Controller (Sect. 4.4)
α
Gait Library (Sect. 4.3.2, 4.3.3)
v_{sag}
Average

Figure 4.1: Feedback diagram illustrating the control structure. The lateral controller is independent of the sagittal plane variables, making it possible to reduce the 3D model to 2D for sagittal controller design. Gait design implements virtual constraints in continuous time. Gait library updates gait parameters step to step based on the current speed, deliberately rendering the closed-loop system "approximately neutrally stable", analogous to a linear inverted pendulum (LIP). The speed regulator works by adjusting swing foot leg angle.

based on [125] and the stepping in place controller is based on [104]. The optimization code used to generate walking gaits is based on [58]. The contribution lies in the methods for gait transition and speed regulation, which are achieved by inducing the longitudinal velocity of the robot to approximately evolve step to step as a discrete-time integrator, analogous to that of a linear inverted pendulum [60, 98, 99], and then stabilizing the integrator with a proportional-derivative foot-placement algorithm. The controller is illustrated in Fig. 4.1.

4.3.1 Gait Design Using Virtual Constraints

Virtual constraints are kinematic relations among the generalized coordinates of the robot that are enforced asymptotically via continuous-time feedback control instead of by external forces. One virtual constraint in the form of a spline is imposed for each independent actuator in the planar model. Parameter optimization is used to select the coefficients in the virtual constraints so as to create a periodic orbit achieving a desired walking speed, while respecting physical constraints. Because the planar model has one degree of underactuation, when using the method of virtual constraints for controller design, local exponential stability of a periodic orbit of the closed-loop system can be established on the basis of a scalar quantity that can be directly included in the optimization process [125][pp. 130].

The collection of virtual constraints is expressed as an output vector

$$y = h_0(q) - h_d(s, \alpha), \tag{4.1}$$

to be asymptotically zeroed by a feedback controller. Here, $h_0(q)$ specifies the quantities to be controlled

$$h_0(q) = \begin{bmatrix} q_x \\ q^{st}_{KA} \\ q^{sw}_{LA} \\ q^{sw}_{KA} \end{bmatrix}, \tag{4.2}$$

where st and sw designate the stance and swing legs, respectively, and $h_d(s, \alpha)$ is a 4-vector of Beziér polynomials in the parameters α specifying the desired evolution of the $h_0(q)$.

When the desired walking speed of the periodic orbit is non-zero, the gait phasing variable, s, is defined as

$$s := \frac{\theta - \theta_{init}}{\theta_{final} - \theta_{init}} \in [0, 1], \tag{4.3}$$

where θ is the absolute stance leg angle defined in Fig. 3.2(c), θ_{init} is the initial value of θ each step, and θ_{final} is the final value of θ corresponding to the periodic orbit found in optimization. When the desired velocity is zero, or below a few tenths of a meter per second, time, normalized between zero and one, is used instead to parameterize the gait:

$$\tau := \frac{t}{T} \in [0, 1] , \tag{4.4}$$

where T is the duration of a step. In this case,

$$y = h_0(q) - h_d(\tau, \alpha). \tag{4.5}$$

In both cases, the optimization is performed as in [125] [Chp. 6.6.2], with the constraints given in Table 4.1. Technically, when using (4.5), the model is augmented with

$$\dot{t} = 1, \tag{4.6}$$

and the zero dynamics are computed for the augmented model, and instead of integrating the squared torque and normalizing by step length [123][eq. (46)], the cost is taken as

$$J = \int_0^T ||u(t)||_2^2 dt, \tag{4.7}$$

with T fixed at 0.35 s [104]. Time-based gaits were designed for linear velocities in the sagittal plane, v_{sag}, for the range $-0.4 \leq v_{\text{sag}} \leq 0.2$ in increments of 0.2 m/s, and phase-based gaits were designed for $0.4 \leq v_{\text{sag}} \leq 0.8$, with the same increments.

4.3.2 Gait Library and Locally Stabilizing Feedback Controller

Four time-based gaits were designed for $-0.4 \leq v_{\text{sag}} \leq 0.2$ in increments of 0.2 m/s and four phase-based gaits were designed for $0.4 \leq v_{\text{sag}} \leq 1.0$ with the same incre-

Table 4.1: Optimization constraints

Motor Torque $\lvert u \rvert$	< 5 Nm
Impact Impulse Fe	< 20 Ns
Friction Cone μ	< 0.4
Vertical Ground Reaction Force	> 300 N
Mid-step Swing Foot Clearance	> 0.15 m

ments, for a total of eight gaits[3]. For values of v_{sag} between the discrete speeds, $v_{\text{sag,i}}$, $1 \leq i \leq 8$, define the Beziér coefficients by linear interpolation

$$\zeta(v_{\text{sag}}) = \frac{v_{\text{sag}} - v_{\text{sag,i}}}{v_{\text{sag,i+1}} - v_{\text{sag,i}}}, \ 1 \leq i \leq 7 \tag{4.8}$$

$$\alpha(v_{\text{sag}}) = (1 - \zeta(v_{\text{sag}}))\alpha_i + \zeta(v_{\text{sag}})\alpha_{i+1}. \tag{4.9}$$

The collection of gait designs is denoted by

$$\mathcal{A} = \{\alpha(v_{\text{sag}}) \mid -0.4 \leq v_{\text{sag}} \leq 1.0\}. \tag{4.10}$$

On the 2D and 3D models, as well as the robot, the virtual constraints were implemented as in [123, Eqn. (47)] and [49] with a PD controller

$$u = -H^{-1}(K_p y + K_d \dot{y}), \tag{4.11}$$

where

$$H = \begin{bmatrix} -1/2 & -1/2 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

[3]The number of gaits is arbitrary. A finer grid did not change the results. A coarser grid was not tried.

is a constant matrix converting from control coordinates in (4.2) to the actuated coordinates. The gains K_p and K_d are the same for each gait.

4.3.3 Longitudinal Speed Regulation

The objective is to develop a speed controller with a broad basin of attraction. Tracking of a desired walking speed from zero to 0.8 m/s will be obtained. The design is done on the 2D model and then implemented on the 3D model.

Let $v_{\text{sag}}[k]$ be the average longitudinal speed of the robot at the middle of step k. The parameters of the virtual constraints (4.1) or (4.5) are updated to

$$\alpha[k] = \alpha(v_{\text{sag}}[k]). \tag{4.12}$$

This has two important consequences:

1. The local sagittal gait controller being applied is the one optimized for the current walking speed. Hence, the current controller respects the constraints enumerated in Table 4.1. In particular, impact impulse, a key source of failure in experiments, should respect the designed constraints.

2. The closed-loop system is now "approximately neutrally" stable with respect to walking speed, similarly to an inverted pendulum or (IP) [60, 67, 93], or more generally, what has been called a symmetric hybrid system (SHS) in [98, 99]. The equilibrium point of the closed-loop system now corresponds to $v_{\text{sag}}[k]$. If at the next step the robot's speed is perturbed by δv, under the update policy (4.12), the system's new equilibrium point will correspond to $v_{\text{sag}}[k+1] = v_{\text{sag}}[k] + \delta v$. This is similar to an integrator or a LIP model which has an eigenvalue exactly at one. Moreover, the eigenvalue of the closed-loop system that has been deliberately placed near one is independent of the current walking speed of the robot.

It is checked in Sec. 4.5 that this "re-centering" of the gait parameters about the current walking speed leaves all of the eigenvalues but one strictly within the unit circle. Hence, the advantages of the controllers designed above are maintained. Moreover, because of the approximate neutral stability of longitudinal speed step-to-step, a PD foot-placement controller can achieve speed regulation and tracking over a wide range of speeds with a constant set of controller gains.

The longitudinal speed regulator functions very similarly to the lateral controller in Sect. 4.4. Specifically, it sets a target offset in the swing leg angle, $q_{LA} := \frac{1}{2}(q_1+q_2)$, for the non-stance leg, via

$$\begin{aligned} \delta^{sw}_{LA,tgt}[k] =& K_p\left(v_{\text{sag}}[k] - v^{\text{ref}}_{\text{sag}}\right) \\ &+ K_d\left(v_{\text{sag}}[k] - v_{\text{sag}}[k-1]\right), \end{aligned} \tag{4.13}$$

where $v^{\text{ref}}_{\text{sag}}$ is the reference speed. The offset can be implemented in many ways. When using virtual constraints, it can simply be added to the last value of the Beziér coefficients for the swing leg angle. Detailed virtual constraint based implementation could be seen in [91, 124].

4.3.4 Unifying The Timing Variable or Phase Used for Control

In (4.5), the links of the robot and its posture are being coordinated on the basis of time, while in (4.1) they are being coordinated with respect to an internal phase variable. Time- and phase-based parameterizations have relative advantages in different ranges of speed. When the robot is stepping in place for example, the phase variable (4.3) is not even well defined because θ_{init} and θ_{final}, the initial and final values of the mechanical phase variable, are approximately equal. In such a case, time is a more appropriate variable for synchronizing leg motion. When the robot is moving slowly, step length is very short and $\theta_{final} - \theta_{init}$ is only a few degrees, which may introduce too much sensitivity into the gait parametrization. On the other hand, extensive experimentation on Rabbit, Ernie [75], MABEL, and Amber [6] has shown the robustness of the phase-based implementations when walking at speeds exceeding several tenths of a meter per second. Gregg et al. are compiling evidence that human response to disturbances may actually better correlate with a mechanical phase variable than to time [47]. Kong et al. show that iterative learning control (ILC) performs better when trajectories are parameterized using phase rather than time [66].

The following is a means to combine time- and phase-based parameters into a single quantity. Let T be a nominal period of a gait and define

$$\dot{\tau} = \frac{1}{T} + \frac{L}{T}\left[s(q) - \tau\right], \ \tau^{+} = 0 \tag{4.14}$$

where $s(q)$ is computed as in (4.3) and $L \geq 0$ is a gain to be chosen. This can be viewed as a Luenberger observer for the phase. When $L = 0$, (4.14) reduces to (4.4), and a singular perturbation argument shows that as $L \to \infty$, (4.14) is purely phase-based. Here it is selected as

$$L(v_{\text{sag}}) = \begin{cases} 0 & |v_{\text{sag}}| < 0.4 \\ 7v_{\text{sag}} & \text{otherwise} \end{cases}, \tag{4.15}$$

which is sufficient to regard τ as "approximately" purely phase-based when $v_{\text{sag}} \geq 0.7$ m/s. Note that the feedforward-term, $\dfrac{1}{T}$ remains when $L > 0$, and the value used here is $T = 0.35$.

4.4 Lateral Control for Straight-line Walking

While the design of the lateral controller is not a focus of this work, we provide in this section a lateral foot-placement strategy, based on [104], that is sufficient to extend 2D controllers to 3D for the robot MARLO.

On a step-to-step basis, the objective is to obtain an asymptotically stable relationship

$$v_{\text{lat}}[k+1] = -k_1 v_{\text{lat}}[k] - k_0 v_{\text{lat}}[k-1], \tag{4.16}$$

where k is the current step index, $v_{\text{lat}}[\ell]$ is average lateral velocity at step ℓ, and k_1, k_0 are gains such that the roots of $\lambda^2 + k_1\lambda + k_0$ are in the unit circle. Foot placement is a common means of regulating velocity over the ensuing step through selection of the ground contact point of the inverted pendulum formed by the stance leg end and the center of mass. A strategy based on the average lateral velocity has the benefit of working independently of sagittal control, potentially enabling a 3D walking platform to function for a set of 2D controllers, as illustrated in Fig. 4.1.

Following [104, Eqn. (5)], lateral foot placement is regulated by the target swing hip angle by

$$\begin{aligned} q^{sw}_{3,tgt}[k] =& q_{3,\text{nom}} + K_p\left(v_{\text{lat}}[k] - v^{\text{ref}}_{\text{lat}}\right) \\ &+ K_d\left(v_{\text{lat}}[k] - v_{\text{lat}}[k-1]\right), \end{aligned} \tag{4.17}$$

where $q_{3,\text{nom}}$ is the nominal hip angle. The gains K_p and K_d are tuned by hand. Because it is the contact point where the foot is placed that matters, as Kuo points out in [70], any continuous-time controller that achieves $q^{sw}_{3,tgt}[k]$ before ground contact should be sufficient to stabilize the lateral dynamics. For normal forward walking and stepping in place, $v^{\text{ref}}_{\text{lat}} = 0$, although a non-zero reference velocity can be used to achieve sideways walking.

Torso roll angle stabilization is achieved using stance hip actuation as explained in [104, Eqn. (10)].

Table 4.2: Largest eigenvalue of various controllers

v (m/s)	**0.3**	**0.5**	**0.7**
2D VC	0.93	0.94	0.91
2D VC-SR	0.76	0.73	0.64
3D L-VC	0.91	1.12	1.14
3D L-VC-SR	0.86	0.82	0.81

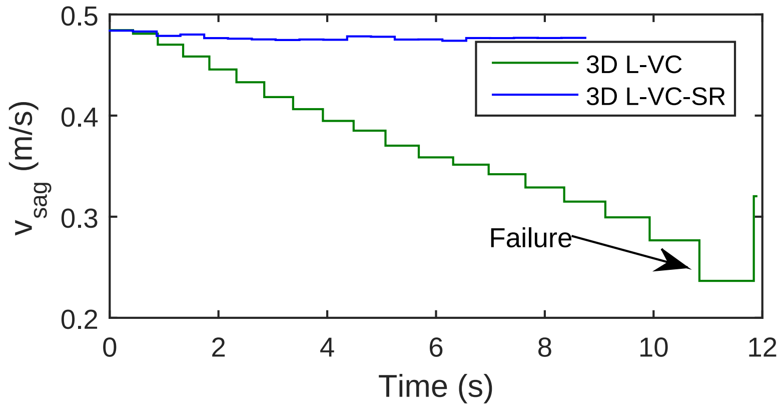


Figure 4.2: Virtual constraints are designed for 0.5 m/s 2D walking and verified in 3D with the lateral controller. The controller (3D L-VC) slows down and fails, whereas the speed regulator (3D L-VC-SR) maintains the desired speed.

4.5 Model-based Analysis and Simulation

This section provides analytical and simulation support for the various components of the overall controller presented in Sects. 4.3 and 4.4.

4.5.1 Local Exponential Stability in 2D and 3D

The Poincaré map was used to check the existence of fixed points (i.e., periodic orbits) and to evaluate their stability. Table 4.2 first shows that the interpolated virtual constraint controllers (4.8), labeled (**2D VC**), at 0.3, 0.5 and 0.7 m/s induce

Table 4.3: Three largest eigenvalues when $\alpha[k]$ is updated

v **(m/s)**	**0.3**	**0.5**	**0.7**
	1.04	0.98	1.09
2D VC-U	0.26	0.31	0.39
	0.19	0.20	0.25

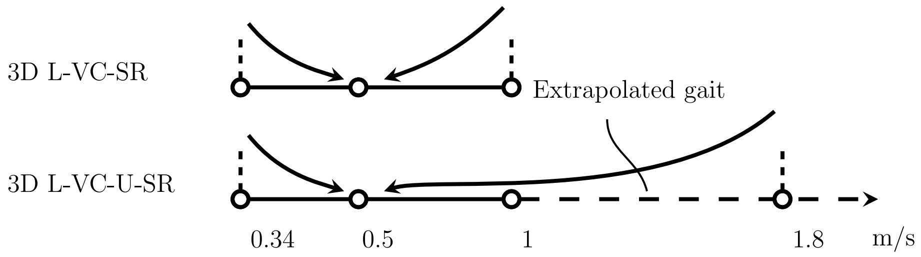


Figure 4.3: Updating $\alpha[k]$ enhances ability to reject velocity perturbations. The Figure is a cartoon of a basin of attraction; it is not to scale.

local exponential stable fixed points in the 2D model, and that including the sagittal plane step length regulator (4.13), labeled (**2D VC-SR**), reduces the magnitude of the largest e-value. When implemented on the 3D model in combination with the lateral controller (4.17), stability is not always maintained for the virtual constraint controller, labeled (**3D L-VC**), as shown in row 3 of Table 4.2, while the inclusion of the step-length regulator achieves local exponential stability; see (**3D L-VC-SR**) and row 4 of Table 4.2. It is important to note that in this analysis, the parameters in the virtual constraints are being held constant step to step; they are not being updated as in (4.12).

Simulations of the (**3D L-VC**) and (**3D L-VC-SR**) controllers for 0.5 m/s are shown in Fig. 4.2. Without explicit speed regulation, the robot with (**3D L-VC**) slows to a point that it cannot complete a step and falls.

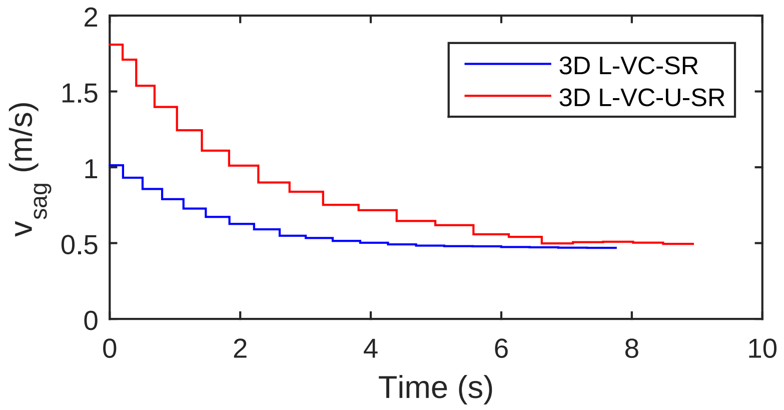


Figure 4.4: Two controllers recovering to nominal speed 0.5 m/s from respective largest speed perturbation.

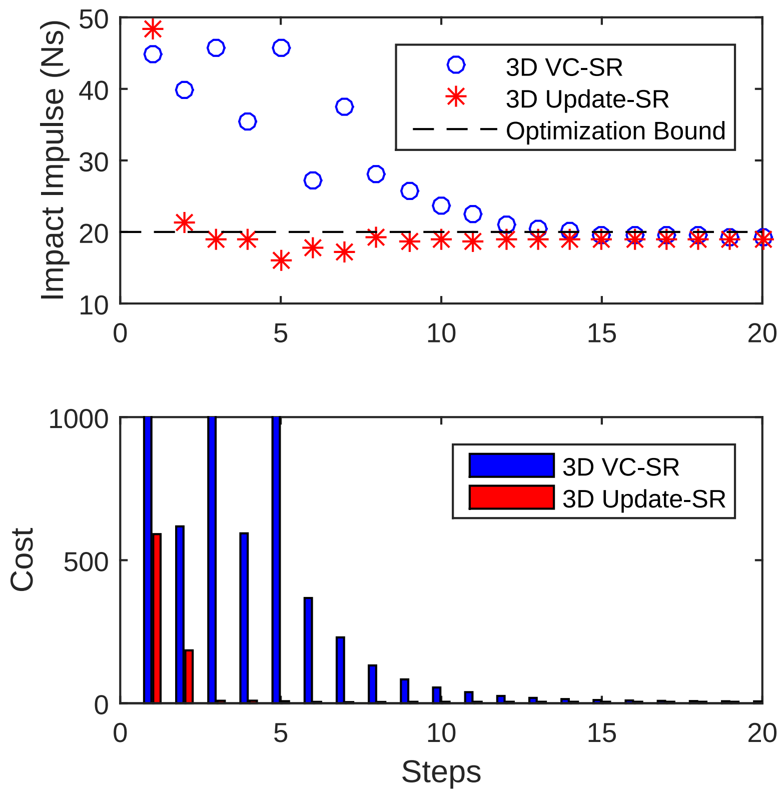


Figure 4.5: (Top) Evolution of impact impulse optimization constraint with respect to step number when parameter updates are used versus not used. (Bottom) Associated cost function on the torques. Both simulations are initialized at 1 m/s and settle to the nominal speed of 0.5 m/s over 20 steps. Parameter updates significantly reduce the impulse during ground contact and the optimization cost (4.7).

4.5.2 Parameter Update based on Longitudinal Speed

Table. 4.3 illustrates the “approximate neutral” stability of the planar controller, labeled (**2D VC-U**), when the gait parameters are updated step to step on the basis of longitudinal speed (4.12). It is seen that the largest eigenvalue is near one where others are significantly less than one.

By updating the controller parameters, the gait design is “re-centered” at the current operating point of the robot. Some of the benefits of performing the parameter updates are now illustrated. Figure 4.3 shows improved ability to handle longitudinal speed perturbations in 3D. When the controller parameters were initialized at $v_{\text{sag}} = 0.5$ m/s and held constant, the **3D L-VC-SR** controller can recover from an initial velocity of 1.0 m/s, but fails for larger speeds. On the other hand, when controller parameters were initialized at $v_{\text{sag}} = 0.5$ m/s and updated step to step, **3D L-VC-U-SR**, the simulated closed-loop could recover from an initial velocity of 1.8 m/s, considerably beyond the design range of the controller library[4]. A plot of speed versus time is shown in Fig. 4.4 over 20 steps of the gait.

A more important benefit is illustrated in Fig. 4.5, namely the reduction of the swing leg impact impulse[5] and a reduction in the cost (4.7). Very soon after the first parameter update takes place, the impact impulse constraint given in Table 4.1 is respected. On the other hand, without parameter updates, is only asymptotically respected.

4.6 Experimental Results

This section illustrates several of the controllers on the physical 3D robot. Videos are available at Extension 2 and 3.

[4]Linear extrapolation was used to define the controller parameters.
[5]Related to swing leg vertical velocity at impact; see [87] for its correct definition.

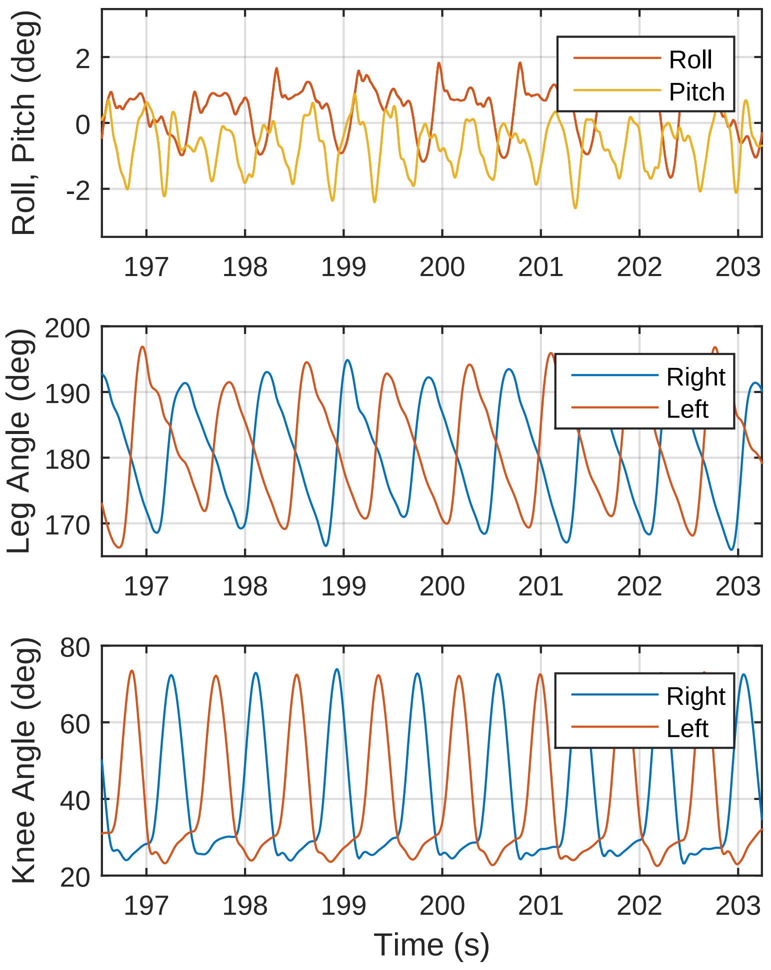


Figure 4.6: Configuration variables changed during walking around 0.8 m/s. Roll was controlled to zero by lateral controller, while other variables were virtually constrained by sagittal controller.

4.6.1 Partial Controller

The controller (**3D L-VC**) was executed five times on the robot, which was initialized at 0 m/s (i.e., stepping in place), speed was increased to 0.4 m/s, and then the controller parameters were frozen at $v_{\text{sag}} = 0.6$ m/s. It fell quickly on four of the trials and achieved ten additional steps on one trial. This seems to support the analysis in Table. 4.2.

When the speed regulator is added, the controller (**3D L-VC-SR**) produced consistent outcomes, with the robot always walking the full distance of the laboratory, approximately 10 m.

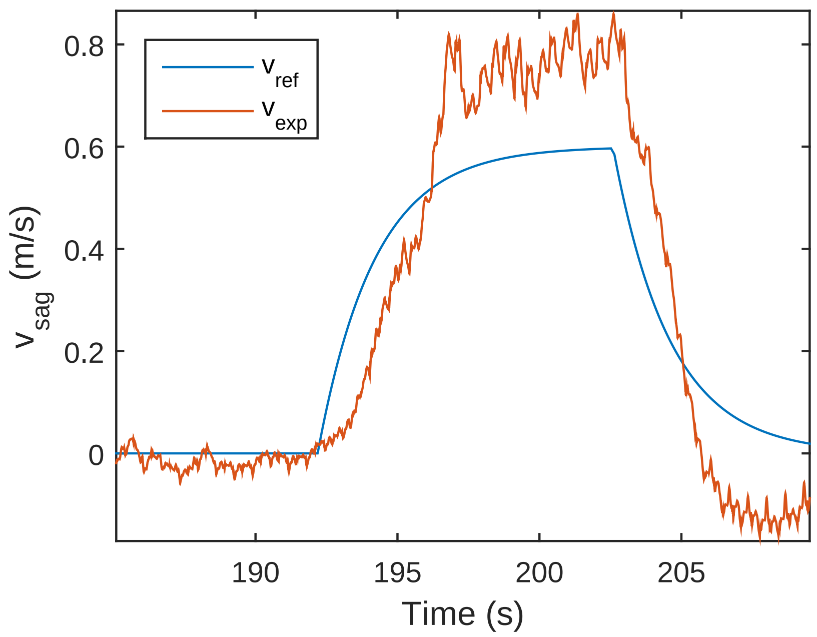


Figure 4.7: Gait transition under complete controller. The robot transitions from zero to 0.8 m/s following the reference speed and then returns to stepping in place.

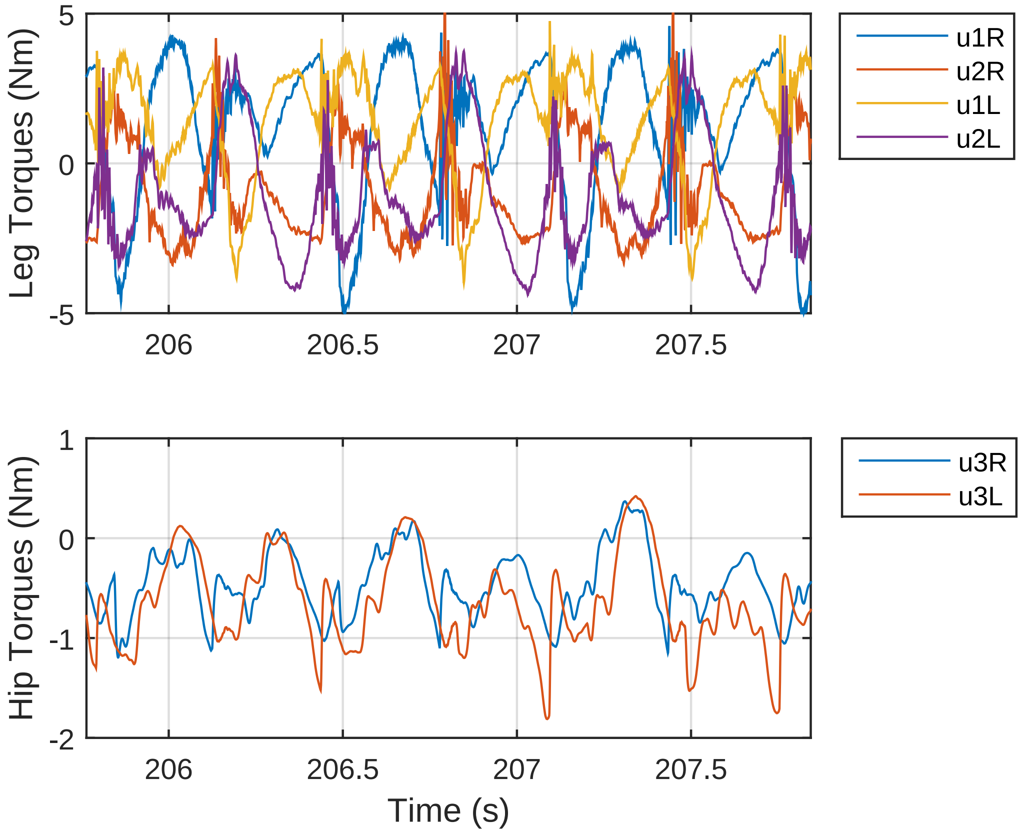


Figure 4.8: Torque output of actuators on the motor side during stepping in place. The leg motors are connected with a 50:1 gear box while the hip's gear ratio is 26.7:1

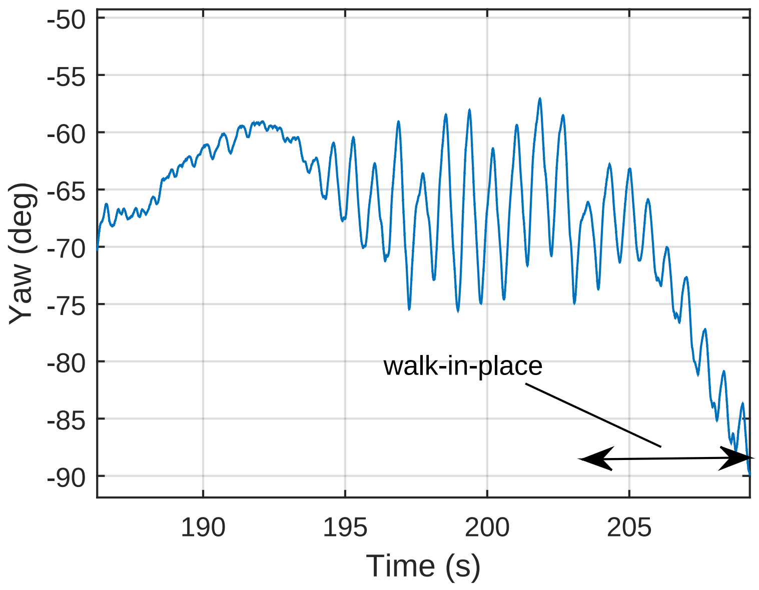


Figure 4.9: Although the two-point-contact feet reduce the yaw motion, it is not fully eliminated. Yaw angle drifted about 20° while stepping in place.

4.6.2 Complete Controller of Figure 4.1

The complete virtual constraint controller with updated gait parameters and speed regulator (**3D L-VC-U-SR**) is implemented on indoor and outdoor experiments. The controller achieves a top walking speed of approximately 0.8 m/s 2. The longest walk in a single experiment is 260 meters with ±7 degrees of slope variation 3. The posture of the robot, as reflected in the pitch and role angles of the torso, are consistently maintained as shown in Fig 4.6; in addition, it was consistent performance across multiple experiments. Compared to aforementioned partial controllers, the complete controller is able to transit seamlessly from 0 m/s to a target speed and then return to stepping in place as shown in Fig. 4.7. To achieve this transition, **3D L-VC-U-SR** uses (4.14) and (4.15) to coordinate time- and phase-based control. Low speed walking uses time-based synchronization while walking speeds above 0.4 m/s transition to phase-based synchronization.

3D L-VC-U-SR is able to walk in place with lower torques than previous work with ATRIAS in [104] (see Fig. 4.8). Lower torques are the result of knees that are straighter and more rigid like the robot Rabbit [23], a torque-based optimization criterion (4.7), and constraints to avoid saturation (Table. 4.1).

Fig. 4.9 shows yaw angle over a transition experiment. The feet successfully reduced yaw compared with previous work with ATRIAS in [21][Fig. 3], but yaw is not completely constrained. During stepping in place, yaw drifts 20°.

4.7 Discussion and Conclusions

This chapter approached 3D walking by designing separate control modules for lateral stabilization, sagittal gait design, and longitudinal speed regulation, as shown in Fig. 4.1. An advantage of this approach is that it gives a hierarchial view of control design. The lateral controller reduced the 3D model to 2D. Advanced control methods could be used efficiently in 2D to design a finite set of stabilizing controllers for a range of fixed walking speeds. In particular, the sagittal plane controllers were designed using a complete mechanical model of the robot, and this allowed important physical restrictions, such as actuator bounds, friction cone, contact impulse, and foot clearance to be directly addressed without hand-tuning on the robot. Interpolation and "re-centering" of the sagittal plane controllers allowed the construction of a continuous gait library of stabilizing controllers. When combined with a linear PD speed regulator, this allowed the tracking of longitudinal speed commands, while continuing to respect important physical constraints even when walking speed varied.

This hierarchical method was shown to be effective on a physical robot by conducting experiments both in the laboratory and outdoors. The robot could be initialized to walk in place, and then commanded to increase speed to 0.8 m/s, followed by a reduction in speed to stepping in place. For these experiments, yaw (or turning) was ignored in the control design. It was assumed instead that a yaw rate of approximately zero would be imposed passively through the foot design.

A clear advantage of using the full 2D model, versus replacing the robot immediately with one of the many pendulum models [60, 96, 95] when doing controller design, was that optimization could be employed, which provided a straightforward means to address the aforementioned physical constraints. A clear drawback of taking a decoupled perspective on the three primary planes of motion, sagittal, lateral, and transversal, is that it limits the type of gaits that can be achieved to those that do not "excite" inherent coupling in the dynamics. Agile motions such as rapid turning [121, 129] or dodging an obstacle clearly require using the full 3D model. For underactuated robots, however, even walking slowly in 3D is challenging because the periods of oscillation in the sagittal and lateral planes must be synchronized [99, 98, 60].

Counting on a physical means for yaw "stabilization" is a leap of faith at best, and a not uncommon means of failure in some of the experiments done on the robot used in this chapter. Active steering is needed and will be added soon.

When stepping in place, the mechanical phase variable used in previous work [125] was not a viable option, so time was used [100, 104]. A Luenberger-like estimator was used to combine time-based and phase-based gait controllers in a seamless manner. In the following work, we will only use the *time-based* gait controller for simplicity and generality reasons.

CHAPTER V

Intuitive Method of Machine Learning Control

This chapter proposes an offline approach to design an explicit model-based feedback control policy using ideas from parameter optimization and Machine Learning (ML). The control design process begins by using parameter optimization to generate both training and testing sets of controllers that induce walking gaits in a bipedal robot model. Virtual constraints provide a convenient parametrization of the feedback control laws and corresponding gaits [125]. The training and testing sets include locally exponentially stable periodic walking gaits at various speeds, both forward and backward, and for various constant ground slopes, flat ground, uphill and downhill. They also include aperiodic gaits that transition among a subset of the periodic gaits in a fixed number of steps.

Supervised learning is then used to train a state-variable feedback control policy. The feature space for the supervised learning includes parameters from a reduced-order biped model (e.g., initial stance leg angle and average speed), exogenous signals (target walking speed is used here, but turning angle could be used as well) and perception input (e.g., terrain height or slope). This policy is compared with a testing set of optimal gaits in simulation and is subsequently evaluated on the 3D underactuated robot MARLO. In a simulation of stepping in place, the learned policy takes at most one more step than an optimal gait to recover from initial velocity and position errors. In experiments, the learned policy allows MARLO to recover from $\approx$ 200 N kick. It also enables MARLO to walk down a 22 deg slope and walk on the Wave Field, which presents sinusoidally varying ground height (see Fig. 5.1).

5.1 Control Policy Overview

The control policy proposed here relates a vector of features to a set of control parameters. This policy will be constructed using supervised learning techniques from a carefully designed training dataset. The process includes:

1. choosing features and control parameters;
2. generating the datasets through optimization;
3. fitting the control policy using a training set with supervised learning algorithms; and
4. assessing the policy with a testing set and simulations.

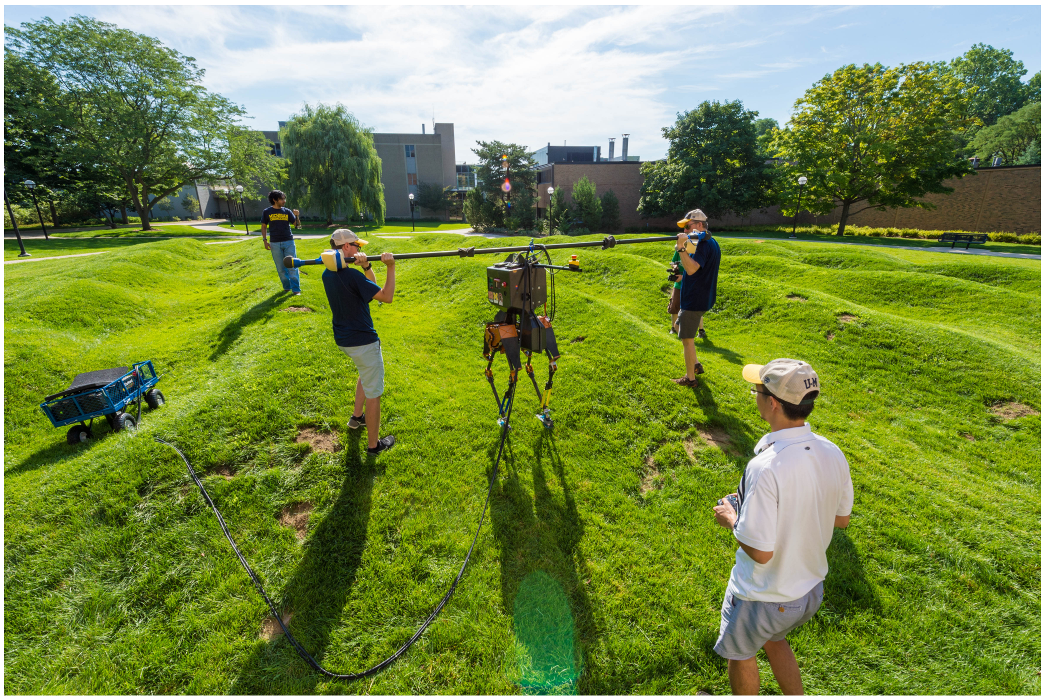


Figure 5.1: Bipedal robot MARLO walked on the University of Michigan's Wave Field, a sinusoidally varying grass terrain. Photo was taken by Roger Hart.

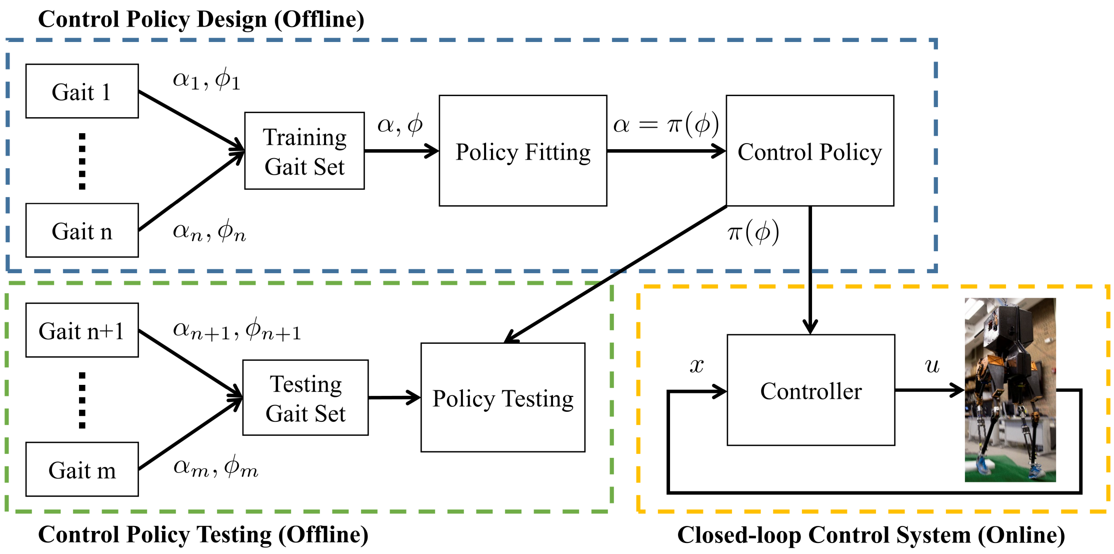


Figure 5.2: Control Policy Design and Implementation

The steps are specified in the following sections for individual policies. Figure 5.2 shows an overview of the policy design process.

5.1.1 Control Policy

A control policy $\pi : \Phi \to \mathcal{A}$ is a function that maps a feature vector $\phi \in \Phi$ to a vector of control parameters $\alpha \in \mathcal{A}$. In this chapter, α is a set of Bézier coefficients inducing a desired trajectory, $q^d(t)$. A low-level feedback controller is then used to minimize the tracking error. The specifics of the feedback controller derivation are given in previous work [32]. The focus of this chapter is to build the control policy.

5.1.2 Dataset Generation Through Optimization

Parameter optimization [125, Sec. 6.3] is used to build a dataset for supervised learning. Each optimization provides a single dynamically feasible path $q^d(t)$ over one or more steps. α and ϕ are extracted at each step. Here, the dataset is constructed from as few as seven to as many as a hundred optimizations, selected to represent the small number of behaviors that the control policy is to learn. All optimizations are

Table 5.1: Optimization constraints

Motor Toque $\lvert u \rvert$	< 5 Nm
Step Duration T	$= 0.35$ s
Friction Cone μ	< 0.6
Impact Impulse Fe	< 15 Ns
Vertical Ground Reaction Force	> 300 N
Mid-step Swing Foot Clearance	> 0.18 m

set up to respect constraints given in Table 5.1 and to minimize the sum of squared torques. Other constraints implemented depend on the nature of the control policy that is to be learned.

5.1.3 Machine Learning Methods

Once the dataset has been generated, various machine learning techniques can be used to regress the control policy $\pi(\cdot)$. This section compares three fitting methods: linear interpolation (LI), support vector machines (SVMs), and neural networks (NNs). The three methods show similar performance in fitting quality and speed tracking. Detailed discussion is available in the simulation section.

5.1.3.1 Linear Interpolation

When the feature ϕ is a scalar, linear interpolation can be used as

$$\pi_{LI}(\phi) = (1 - \zeta(\phi))\alpha_i + \zeta(\phi)\alpha_{i+1} \tag{5.1}$$

$$\zeta(\phi) = \frac{\phi - \phi_i}{\phi_{i+1} - \phi_i}, \tag{5.2}$$

where ϕ_i and ϕ_{i+1} are features in the training set between the input ϕ. It can be extended to bilinear interpolation (BiLI) if the feature has two variables. Since the method only uses local data, it is good to fit an evenly distributed data set.

5.1.3.2 Support Vector Machines

Support vector machines (SVMs) are a common ML technique that can be used for function regression (also known as SVR). The SVM algorithm can be used to regress a nonlinear function by applying the "kernel trick" [18]. In this chapter, the regression was learned using the LIBSVM toolbox with the radial basis function kernel.

5.1.3.3 Neural Networks

Neural Networks (NNs) are an increasingly used method for nonlinear function approximation. They rely on a series of connected "neurons", usually sigmoid functions, and a set of weights that can be learned [34]. In this chapter, the learning is implemented using MATLAB's Neural Network Toolbox with 5 hidden layers. The networks are trained using the default Levenberg-Marquardt algorithm.

5.1.4 Training and Testing

The training and testing datasets are built separately. For each of the learning algorithms listed, a control policy $\pi(\cdot)$ is learned using only the training dataset. Each resulting control policy is assessed using the separate testing dataset. The coefficient of determination (R^2) and the root mean square error (RMSE) provide one way to evaluate how closely the output of the control policy matches the test data. The utility of the control policy is further verified by running simulations.

5.2 Speed Regulation Policy

Chapter IV designed a gait library for speed tracking via optimization. The discrete set of gaits was then interpolated to produce a continuously defined feedback controller. Even when the discrete gaits were (locally) exponentially stable, the resulting closed-loop system was at best neutrally stable. Subsequently, exponential stability was recovered with a supplemental foot placement policy, which allowed MARLO to walk forwards and backwards at a variety of speeds. This section reformulates the design procedure as a supervised learning problem.

5.2.1 Dataset Generation

To generate the training dataset, 13 separate parameter optimizations are run. Each optimization generates a periodic gait at different sagittal velocities v_{avg}. The set of gaits is denoted by

$$\mathcal{A}_{\text{train}} = \{\alpha(v_{\text{avg}}) \mid -1.2 \leq v_{\text{avg}} \leq 1.2\}, \tag{5.3}$$

where v_{avg} increases in steps of 0.2 m/s. A similar testing set of gaits $\mathcal{A}_{\text{test}}$ is designed for the same speed range but at a finer grid of 0.05 m/s. The optimization is set up to respect constraints given in Table 4.1 and to minimize the sum of squared torques. Additional constraints for periodicity and the average velocity are also included.

5.2.2 Feature Selection

The only difference among the optimizations is the average velocity. Therefore, a logical feature choice is $\phi = \{v_{\text{avg}}\}$.

5.2.3 Training Methods

Since ϕ is a scalar quantity, linear interpolation (LI) can be used to fit the control policy $\pi(\cdot)$. This is what was used in [32]. For comparison, support vector machines (SVMs) and neural networks (NNs) are also used.

5.2.4 Stability Remark

When SVM and NNs are used in place of LI, the closed-loop system is also at best neutrally stable. As in [32], this is checked with a Poincaré map.

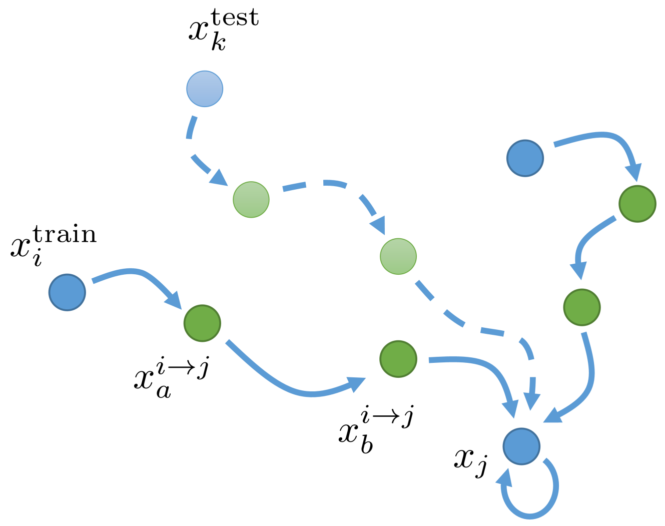


Figure 5.3: A graph of three-step optimization. Given x_i and x_j, the optimization will find a path $x_i \to x_a^{i\to j} \to x_b^{i\to j} \to x_j$ if exists. Blue dots are specified in the optimization while green dots and path are generated from the optimization.

5.3 Transition Gait Policy

The speed regulation policy discussed in the last section "teaches" MARLO how to walk along a steady state, periodic gait. This section proposes an optimization setup to add the transitions between various periodic gaits into the learning process.

5.3.1 Dataset Generation

Let $x_i := [q, \dot{q}]_i^\top$ and $x_j := [q, \dot{q}]_j^\top$ be two points in the robot's state space corresponding to double support. Denote by $\alpha^{x_i \to x_j}$ the control parameters, if they exist, that effect a transition in one step from x_i to x_j. When $x_i = x_j$, we have a periodic gait, and we also denote the control parameters by $\alpha(v^i_{\text{avg}})$, as one of the element in (5.3), inducing a periodic gait at velocity v_{avg}. The corresponding state is denoted by $x^*(v^i_{\text{avg}})$.

To handle a wide range of transitions, we also consider the case where two points in the state space cannot be joined in one step. Specifically, given two points x_i and

x_j, we also design controllers that effect transitions in three steps[1]. Optimization is used to compute two intermediate states $x_a^{i\to j}$ and $x_b^{i\to j}$, and corresponding control parameters, such that, the robot transitions are

$$x_i \to x_a^{i\to j} \to x_b^{i\to j} \to x_j. \tag{5.4}$$

In the language of capture points [12, 94] , x_i above is in the 3-step viable-capture basin of x_j. The 3-step transition gaits are computed for

$$x_j = x^*(v_{\text{avg}}^j), \text{for } v_{\text{avg}}^j \in \{-0.4, -0.2, 0, 0.2, 0.4\} \tag{5.5}$$

$$x_i \in \{x^*(v_{\text{avg}}^i) \mid -0.6 + v_{\text{avg}}^j \leq v_{\text{avg}}^i \leq 0.6 + v_{\text{avg}}^j\}. \tag{5.6}$$

When $v_{\text{avg}}^i = v_{\text{avg}}^j$, it is noted that $\alpha^{x_i \to x_j} = \alpha(v_{\text{avg}}^i)$, the control parameters for the periodic gait at speed v_{avg}^i, given in (5.3). To be clear, each 3-step optimization provides three controllers that are included in the training set.

In this initial study on supervised learning, the testing set focuses on stepping in place. The three-step optimization process as in (5.4) is used to compute controllers given the terminal point $x_j = x^*(0)$ (stepping in place) and initial points x_i that has perturbations of stepping in place. These perturbations correspond to the robot being in double support, in which the support leg angle θ is perturbed $\pm 15\,\text{deg}$, the support leg angle rate $\dot{\theta}$ is perturbed $\pm 34\,\text{deg}/s$, and the swing leg angle rate $\dot{q}_{LA}^{sw}$ is perturbed $\pm 114\,\text{deg}/s$, all independently.

5.3.2 Feature Selection

In the speed regulation policy design, the only changed optimization constraint is the average speed. This led to a logical choice of the feature being v_{avg}. In contrast,

[1]The number three is motivated by [131]. In case the transition can be done in two steps, $x_b^{i\to j} = x_j$; similarly for one step.

when optimizing transition gaits, all of the states change. The feature vector could potentially use the full states, but this may require a large training dataset. Instead, a small set of features $\phi = \{v_{\text{avg}}, \theta_{\text{init}}, v_{\text{tgt}}\}$ is proposed. Inspired from the inverted pendulum model, these features capture the two crucial underactuated degrees of freedom as well as the target velocity. Kernel principal component analysis (PCA) may be used in the future to find a low dimensional representation of the state space to extract features from.

5.3.3 Training Methods

The transition control policies are trained[2] using SVMs and NNs. These policies are assessed by simulating from the initial states x_i in the testing dataset. The simulation results are compared against the optimized gaits in the testing dataset.

5.3.4 Stability Remark

For periodic gaits, current and desired speed are identical. For transitioning among periodic gaits, or when rejecting a perturbation, these two speeds are different which is why we design aperiodic gaits. With the richer feature set $\{v_{\text{avg}}, v_{\text{tgt}}\}$, and with the richer training set, {periodic, aperiodic}, Poincaré analysis verifies that (local) exponential stability is recovered.

5.4 Terrain Adaption Policy

This section adds periodic gaits for different terrain heights or slopes to design a terrain adaption policy. It will enhance the speed tracking performance on sloped terrain and robustness over uneven terrain.

[2]Because we are fitting a small set of features to the data when training the policy, we are not using interpolation. SVMs and NNs are performing regression on the data.

Since the MARLO does not have any vision sensors to foresee the terrain, proprioceptive sensors are used to measure the positions of the feet in the double support phase to estimate the terrain profiles.

5.4.1 Dataset Generation

To generate the datasets, a 2D grid of gaits is optimized. The training dataset includes gaits where v_{avg} ranges [-1.2, 1.2] m/s in 0.2 m/s steps, and h ranges [-0.1, 0.1] m in 0.05 m steps. The testing dataset is designed on the same range but at a finer grid, 0.1 m/s increments for v_{avg} and 0.02 m increments for h.

5.4.2 Feature Selection

5.4.2.1 Sagittal Terrain Adaption

Since the dataset was generated using varying velocities and step heights, the empirical choice for the feature vector is $\phi = \{v_{\text{avg}}, h\}$. The controller is designed using the planar model, thus h is measured as sagittal terrain height.

5.4.2.2 Lateral Terrain Adaption

In the 3D model, the feet height and side width in the double support phase can be also used to estimate the lateral terrain slope β_{lateral}. This chapter shows the preliminary use of this feature in the experiments Section 5.6.3. More sophisticated terrain profile estimation could be used, though the design of control policy remains the same.

5.4.3 Training Methods

Since the dataset was constructed uniformly on a grid, a bilinear interpolation can be implemented. More advanced regression methods (SVMs, NNs, etc.) could be used, but the authors did not pursue them for this control policy. However, a unified control policy, which is described in Section 5.6.4, is fit using a neural network. It combines terrain adaption with speed regulation and transition gaits

5.5 Simulation

The control policies are evaluated in two ways: comparing the control parameters with the testing data, and assessing the control performance in simulated planar walking.

5.5.1 Speed Regulation Policy

The speed regulation policy is generated by three regression methods: π_{LI}, π_{SVM} and π_{NN}. The elements in α show a strong correlation with the testing data in Fig. 5.4, where the lowest coefficient of determination R^2 is 0.9 and the biggest root mean square error (RMSE) is 0.5 deg. These 30 elements are 5 sets of Bézier coefficient that induce $q^d(t)$. The biggest RMSE error of $q^d(t)$ between the control policies and optimization is 0.4 deg, where the position tracking error in experiments is 5 deg on average. The control policies are subsequently evaluated by tracking a target velocity in Fig. 5.5. The three methods give consistent results indicating that the supervised learning approach proposed in this chapter is not limited to a certain method. Small speed tracking error comes from a low-level feedback controller.

5.5.2 Transition Policy

The fitting quality of the transition policy is deteriorated because the features are extracted from a reduced order model in Section 5.3.2, where the biggest RMSE is 8 deg. More discussion will be included in the next chapter. The control policies are analyzed by simulating from the three largest initial state perturbations in the testing dataset $\{\delta\theta = -15 \text{ deg}, \delta\dot{\theta} = 34 \text{ deg}/s, \delta\dot{q}^{sw}_{LA} = -114 \text{ deg}/s\}$, shown in Fig. 5.6. The three-step optimization gives optimal controllers that converge within

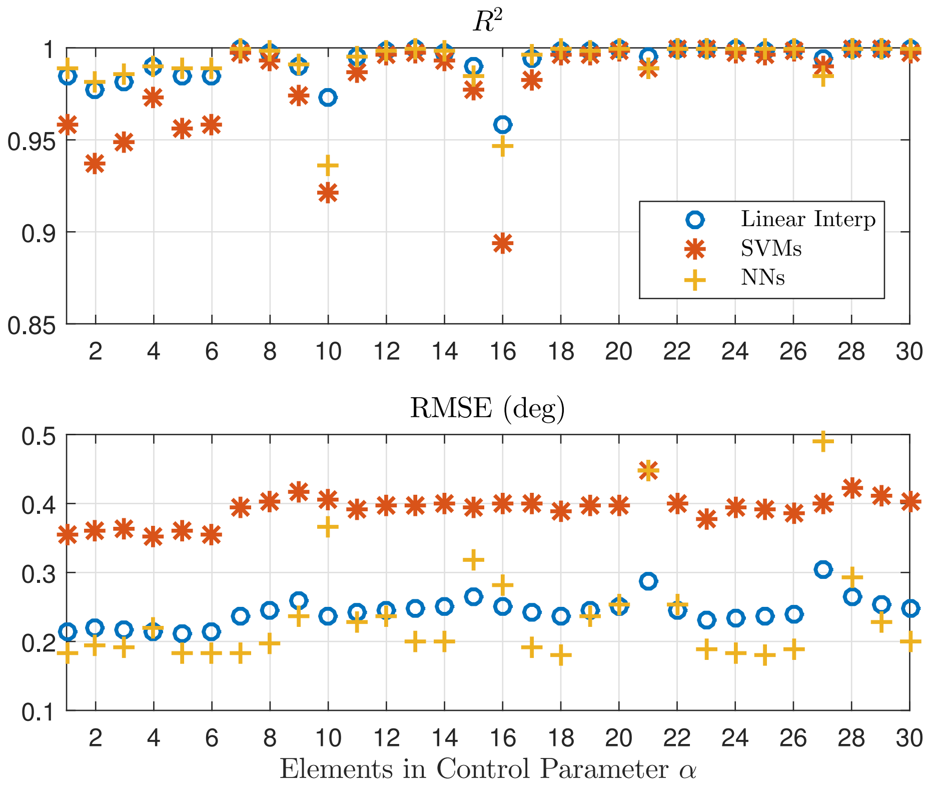


Figure 5.4: Fitting quality of each element in control parameters α of the Beziér polynomial. Because there are five joint trajectories being fitted (torso, stance leg, stance knee, swing leg and swing knee), and each trajectory uses a fifth order Beziér polynomial (i.e., 6 coefficient), there are 5x6 = 30 elements.

three steps. The control policy learned from the training set takes at most one more step to recover.

The transition policy is compared with the speed regulation policy through a perturbation rejection test. The push force is 200 N in one step (0.35 s), shown in Fig. 5.7. The transition policy converges back to the target velocity faster and with less overshoot.

5.5.3 Terrain Policy

Since the training set includes gaits that function correctly for sloped ground, the terrain adaption policy improves speed regulation on both uphill and downhill walking. In Fig. 5.8, both the speed regulation policy and terrain adaption policy

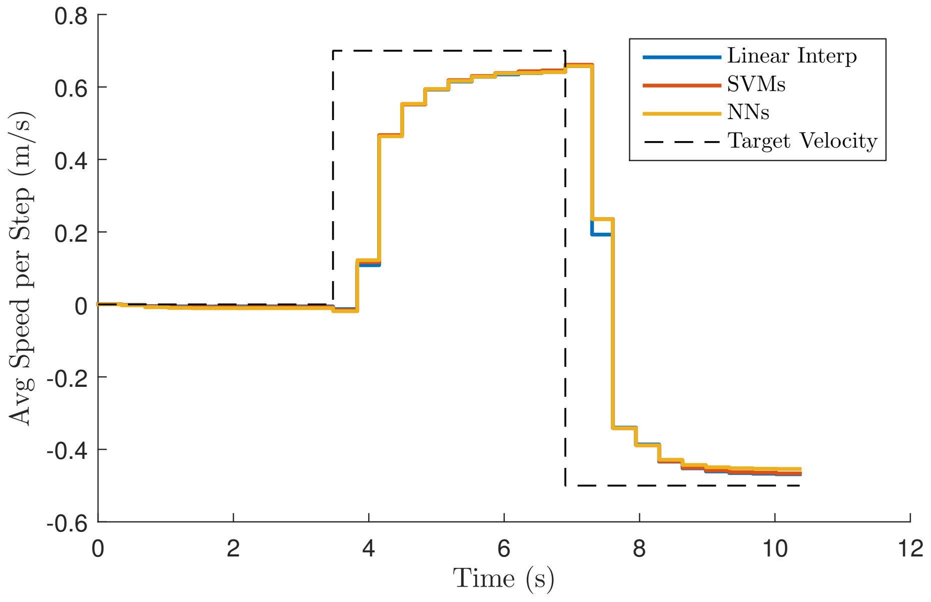


Figure 5.5: All three fitting methods show consistent speed tracking performance indicating the fitting method is not limited to any specific one.

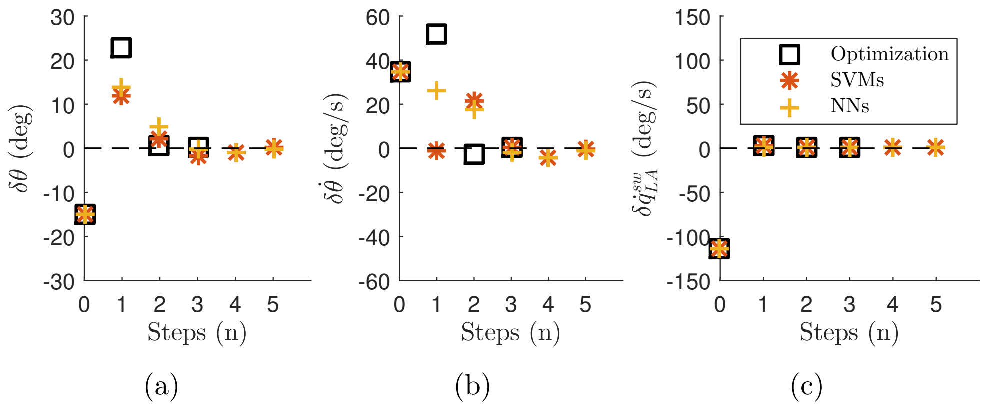


Figure 5.6: (a) has 15 deg initial error on θ. (b) has 34 deg $/s$ error on $\dot{\theta}$. (c) has 114 deg $/s$ error on q_{LA}^{sw}. The transition policy takes at most one more step than the gaits from optimization to recover from these initial errors.

are applied to walking downhill and uphill. The 10 degree slope is the steepest that the speed regulation policy can handle, though it gains considerable speed going downhill. In contrast, the terrain adaption policy maintains roughly the same velocity throughout.

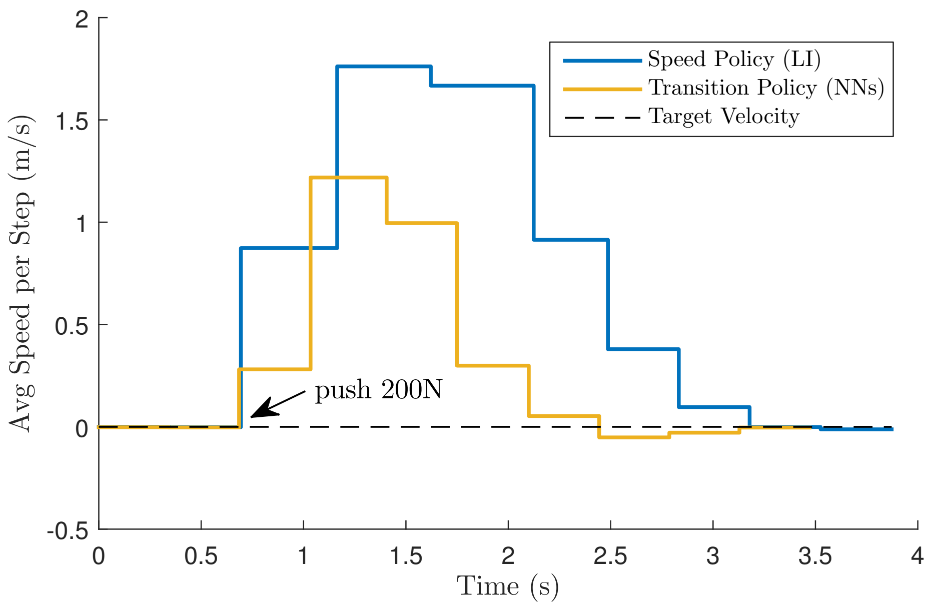


Figure 5.7: After subjecting to a 200N push in one step (0.35 s), the transition policy has shorter settling time and smaller overshoot than the speed policy.

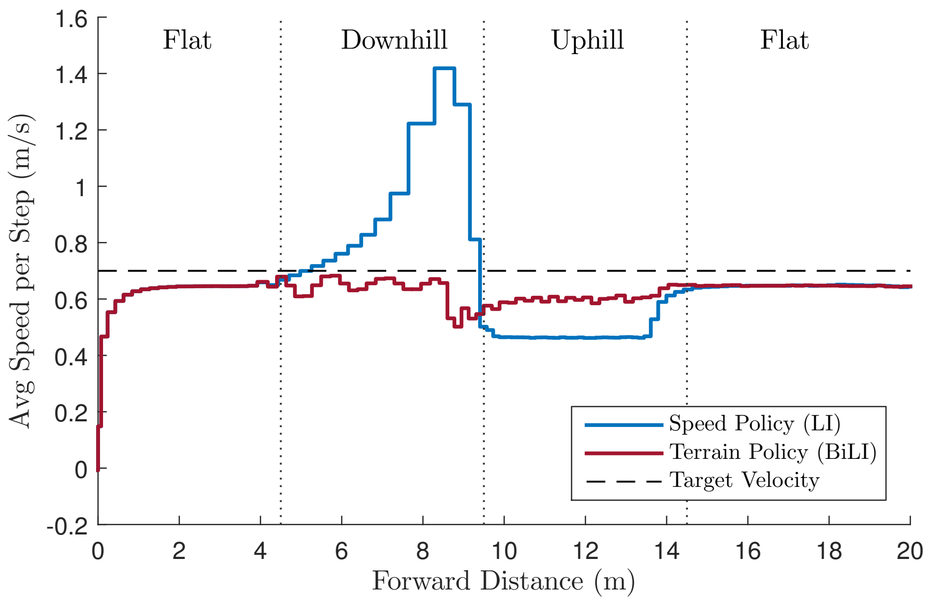


Figure 5.8: The terrain adaption policy chooses a controller based on the current velocity and terrain height, which gives a more consistent speed tracking result than the speed regulation policy. The slope is ±10 deg.

5.6 Experiments and Discussion

For simplicity, the supervised learning policies presented in Sections 5.1 - 5.5 concern the planar model of MARLO. The control polices are augmented with a lateral

Table 5.2: Experiment Videos

Number	Gait Policy	Experiment	Link
1	Speed Regulation [32]	A Long Walk	https://youtu.be/eSllkIptlK0
2	Speed Regulation [32]	Light Push	https://youtu.be/iOltRR0RqiM
3	Transition	Kick MARLO	https://youtu.be/YXJQJtcXX4E
4	Sagittal Terrain	Walking Down 22 Degree Slope	https://youtu.be/gHpXTmyG4mE
5	Sagittal Terrain	Random Terrain	https://youtu.be/iW9SWPQmYh0
6	Sagittal Terrain	Wave Field (First Attempt)	https://youtu.be/YErF0cyPI-g
7	Lateral Terrain	Practice for the Wave Field	https://youtu.be/vEQa1e7lzjQ
8	Lateral Terrain	Wave Field (Second Attempt)	https://youtu.be/TDFz_0Avc2A
9	A Unified Policy	Walking	https://youtu.be/xPHMgFiSeu0
10	A Unified Policy	Pushing, Random Terrain	https://youtu.be/VovWti_wKRU
11	A Unified Policy	Walking in the Forest	https://youtu.be/uYD99f01aek

controller as in [32, 104] for implementation on the physical 3D robot. Controllers for speed regulation were presented in [32]. The experiments for this chapter are numbers 3 - 11 in Table 5.2.

5.6.1 Speed Regulation and Transition

To understand the utility of including the transition gaits in the learning sets, a first control policy is designed using only periodic gaits (see Section 5.2). The asymptotic stability of the closed-loop system is assured with a foot placement controller given in [32, 104]. A second policy is then designed using the same set of periodic gaits augmented with transitions (see Section 5.3), no extra controller is needed. Transition gaits represent transient conditions that have been designed to respect the physical limitations of the robot, and hence the resulting control policy is better able to avoid foot slippage during transients than the policy built on steady-state (periodic) walking. This is demonstrated by comparing the light push in Experiment 2 to the much stronger kick in Experiment 3.

5.6.2 Sagittal Terrain Adaptation

Experiment 1 uses the speed controller, without transitions, discussed above. At the end of Experiment 1, MARLO encounters a 7 deg upward slope, slips, and falls. A new policy is designed that focuses on ground slope changes (see Section 5.4.2.1), and is used in Experiments 4 and 5. Experiment 4 demonstrates the robot walking down a long, 22 deg, steep slope. The average walking speed is about 0.2 m/s; the safety gantry gets stuck at several points, keeping the average speed quite low. Walking up the slope has not been demonstrated because pushing the gantry up a steep hill is impossible. Walking down is often more challenging because the robot will gain speed from gravity. Even though the control policy was designed for constant slopes, in Experiment 5 the robot is challenged to walk indoors over randomly varying terrain. In these experiments, the ground slope is estimated by relative foot height during double support, which is one of the features used in determining the control policy for the next step. If the terrain changes dramatically over a step, a camera is needed to preview the terrain and this information must be added to the feature set during supervised learning.

5.6.3 Lateral Terrain Adaptation

Without a camera, and using only relative foot height information in double support, it is not possible to distinguish between a slope in the sagittal direction, the lateral direction, or a combination. Using the same control policy as in Experiments 4 and 5, the robot was taken to the Wave Field on the University of Michigan Campus; see Experiment 6. The most frequent failure mode was the robot's swing leg hitting the ground prematurely because, when moving the leg laterally, it assumed the slope was zero. A new policy was designed under the assumption the relative changes in foot height are due to a lateral slope only (in the sagittal direction, the ground is assumed flat); see Section 5.4.2.2. Experiment 7 tests the control policy indoors and

Experiment 8 is performed on the Wave Field. In the latter, the robot is able to make two complete passes in the troughs between the crests, whereas in Experiment 6, it never made it more than half way down any one of them.

5.6.4 Unified Policy

Here, the control policy is designed using periodic gaits, transition gaits among a subset of them, and terrain slope changes in the sagittal direction. In Experiment 9, MARLO walks outdoors at speeds varying from standing to 0.5 m/s. In Experiment 10, the robot traverses a pile of rubble in the laboratory. When taken to a a section of woods on the campus in Experiment 11, the robot walks down sloped terrain, covered with branches, and encounters stumps. After about five minutes, MARLO trips on a lateral slope because of the same failure mechanism in Experiment 6: when moving the swing leg laterally, premature impact with the ground occurs.

5.7 Conclusions and Next Steps

Supervised learning was used to design control polices for the complete planar model of an underactuated 3D bipedal robot. The training and testing sets included periodic gaits on flat and sloped ground and transition gaits. The control policy designed with supervised learning increased the robustness of the robot's gait in comparison to previous control solutions that focused on asymptotically stable walking at a constant speed [20], or a solution built by interpolating controllers from a library of such gaits.

Part of the enhanced robustness comes from including transient control solutions in the training set. These provide a means for returning to a target speed after a perturbation, while satisfying constraints on peak torque, friction cone and motor speed. Additional robustness comes from including gaits that functioned correctly on sloped ground. The supervised learning formulation allowed a collection of behaviors to be addressed in a unified manner, when the feature set was expanded to include initial states of a reduced-order model, exogenous command signals, and terrain information gleaned from sensors.

The next chapter will extend the method to address the full 3D dynamic model of the robot. This was not done here because, when the work was initiated, the fast optimizer of Ames's group was not yet available [53]. It will also give the formal way to design the transient gait and the mathematical theory of using Supervised Learning to design controller.

CHAPTER VI

Formal Method of Machine Learning Control

The previous chapters have established two approaches for gait design based on the use of multiple trajectories; one based on a library of periodic gaits and a second method that combined periodic and transient gaits through machine learning. This chapter provides a formal mathematical framework for the use of machine learning in controller design. For the sake of simplicity, let's pretend the model of a bipedal robot can be captured by an ordinary differential equation,

$$\dot{x} = F(x, u), \tag{6.1}$$

with state variables $x \in \mathcal{X}$ and control inputs $u \in \mathcal{U}$. The design process of this chapter begins with the construction of a periodic solution meeting relevant constraints. Denote the period by $T_p > 0$ and the initial condition by ξ^*. The next step is to make an initial selection of a low-dimensional set, let's call it $Z_0 \subset \mathcal{X}$, such that $\xi^* \in Z_0$. Now begins the real work; we seek to design open-loop trajectories of the full-order model

$$(x_\xi(t), u_\xi(t)),\ 0 \leq t \leq T_p,\ \xi \in Z_0 \tag{6.2}$$

that over the interval $[0, T_p]$, "approach" the periodic solution. Specifically, for some $0 \leq c < 1$, and for each $\xi \in Z_0$, we have

$$\varphi_\xi(T_p) \in Z_0 \text{ and } ||\varphi_\xi(T_p) - \xi^*|| \leq c||\xi - \xi^*||. \tag{6.3}$$

After designing the trajectories, we seek to construct a low-dimensional subsystem that realizes them, namely,

$$\begin{aligned} \dot{z}(t) &= G(t, z) \\ x(t) &= H(t, z), \end{aligned} \tag{6.4}$$

with $z \in Z \subset \mathcal{X}$, such that (a) for $\xi \in Z_0$,

$$z(0) = \xi \Rightarrow x_\xi(t) = H(t, z(t)),$$

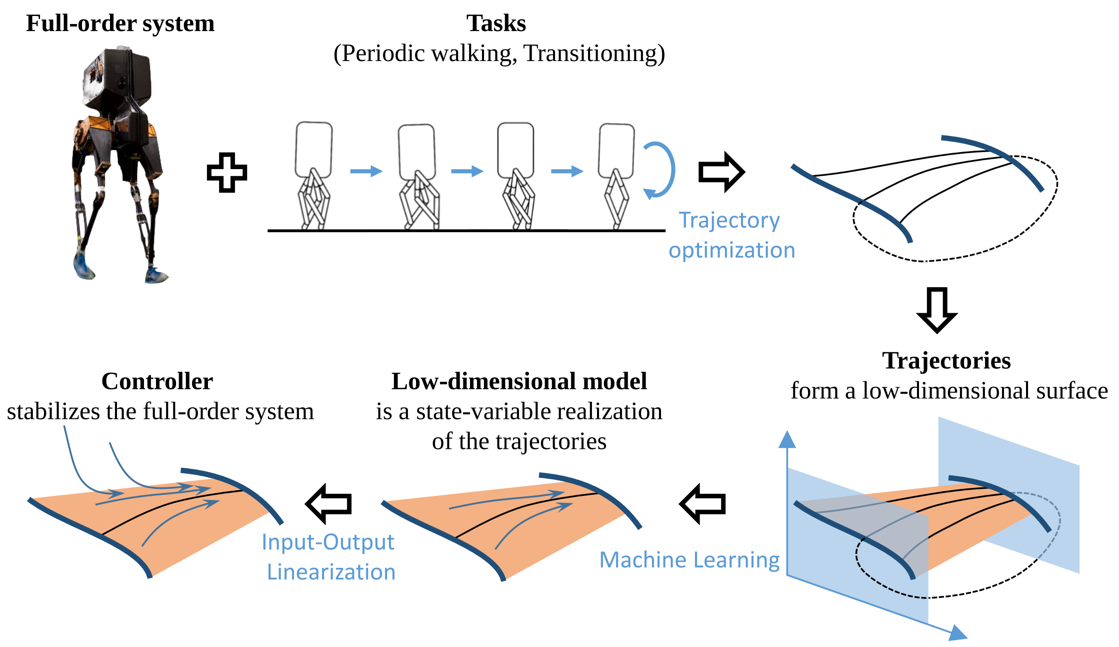
and (b) the periodic motion is a "locally exponentially stable output" of the model. If this can be done, we would argue that (6.4) is a more desirable target model than a typical pendulum because the target has been constructed directly from the full-order model and its "specification", that is, the constraints imposed when designing the trajectories. The overall concept is illustrated in Figure 6.1 and 6.2.

To turn this into a viable feedback design process for the original system, we have to address the following issues:

(i) How to compute the low-dimensional model (6.4) for realistic bipedal robots and the surface[1] $Z \supset Z_0$ on which it is defined?

(ii) As with any low-dimensional target model, how to embed it in the full-order model with stability? In other words, how to design a feedback for the original system (6.1) that does two things: (a) creates an invariant surface Z in its state

[1] Z_0 only contains the initial conditions for the trajectories. The evolution of the trajectories will determine Z. The vector field and output map in (6.4) will be computed through a combination of Supervised Machine Learning and model structure.



Full-order system
Tasks
(Periodic walking, Transitioning)
Trajectory optimization
Trajectories
form a low-dimensional surface
Low-dimensional model
is a state-variable realization of the trajectories
Machine Learning
Controller
stabilizes the full-order system
Input-Output Linearization

Figure 6.1: The Supervised Machine Learning approach.

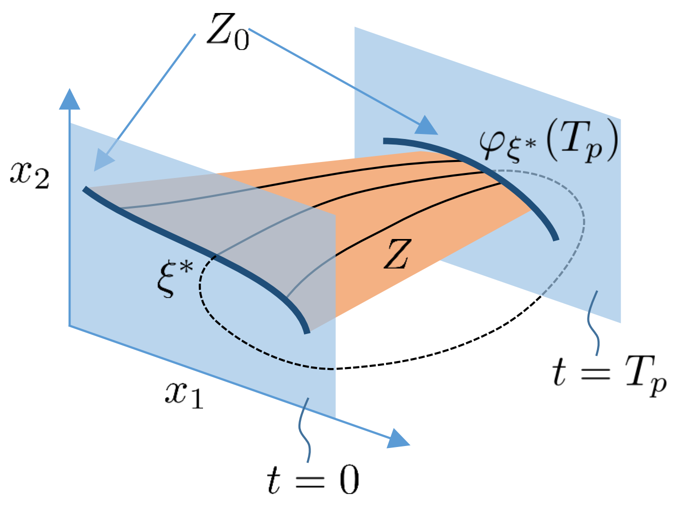


Figure 6.2: Based on model structure, the system's state is decomposed into $x = (x_1, x_2)$, where the dimension of x_1 is much smaller than the dimension of x_2. The surface Z_0, shown here as a line, is the set of initial conditions used to build a set of trajectories that will fill out the surface Z. By construction, this model, if it exists, will be easy to embed in the robot's full state space.

space with restriction dynamics given by (6.4), thereby encoding the desired stable walking motion; and (b) solutions of the closed-loop system starting near the surface asymptotically converge to the surface, thereby realizing the walking motion in a stable manner in the overall system.

The remainder of the chapter is dedicated to addressing these challenges for a class of hybrid models and tasks of interest to bipedal locomotion. Section 6.1 develops the basic ideas in the simpler setting of ordinary differential equations. The results are of independent interest for tasks such as rising from a sitting position or standing in place. Trajectory optimization is used to generate the low-dimensional set of open-loop trajectories (6.2) that includes a metric for attractivity to a periodic solution, a family of periodic solutions, or transitions among such solutions. Model structure and Supervised Machine Learning are proposed as a means to extract functions from the open-loop trajectories to build the low-dimensional model (6.4). Finally, an appeal is made once again to model structure to embed the low-dimensional model in the full-order model while guaranteeing local exponential attractivity.

Section 6.2 illustrates the design process on the well-known inverted pendulum on a cart. This will allow the reader to explore the method on a simple model. Section 6.3 develops the results for hybrid models, preparing the ground for the simulations and experiments reported in Section 6.4 for the bipedal robot MARLO. All proofs are given in Appendix C.

6.1 Presentation of Main Ideas

For the class of robot problems of interest to us, optimal gaits can now be computed in minutes [58, 53], but not in tens of milliseconds, which is what would be required for online use. In the simpler setting of a non-hybrid system, this section develops our main ideas for mitigating the curse of dimensionality in optimization-based controller design. Section 6.1.2 provides the first example of conditions for a family of open-loop trajectories of a model from which a realization can be extracted and its equilibrium will be locally exponentially stable; see also [30, 108]. The process of building the realization from the trajectories is based on regression, namely Supervised Machine Learning. The size of the model that can be treated with these initial results is limited by both the number of optimizations it takes to create the family of open-loop trajectories and the number of features that can be included in the Supervised Machine Learning. Section 6.1.3 extends the design process to building reduced-order target model for a high-dimensional model. Importantly, the design is based on a far smaller set of open-loop trajectories. Sections 6.1.4 and 6.1.5 then provide conditions for embedding the target model in the full-order model such that the origin of the full-order model is locally exponentially stable. The proofs of the results developed in the section are given in Appendix C. The relationship to other controller design methods is addressed in Appendix D.

In presenting the main ideas, we will deliberately organize them as a design philosophy. We choose to let the user rely on his or her wits to meet our conditions, rather than muddying the waters with a set of highly technical sufficient conditions that no one would ever check. We know as well as the readers know that optimization problems are very tricky: it is easy to paint oneself into a corner that only yields non-smooth solutions. On the other hand, many problems, such as the examples worked out in the chapter, seem to have very nicely behaved solutions. We are confident that we did not cherry-pick the only nice problems and that the readers will find a host of further interesting examples.

6.1.1 Model Assumptions

To keep the connection to stabilizing periodic gaits in bipeds, we consider a periodically time-varying nonlinear system with equilibrium point at the origin,

$$\dot{x} = f(t, x, u). \tag{6.5}$$

The coordinate transformation required for shifting a periodic solution of a nonlinear model to the origin is provided in [63, pp. 147]. An equilibrium point is treated as a special case of a periodic orbit where the period can be any number $T_p > 0$; in particular, when discussing periodicity, T_p is *not required to be a fundamental period.*

The ODE (6.5) is assumed to satisfy the following conditions.

A-1 $f : [0, \infty) \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is locally Lipschitz continuous in x and u, piecewise continuous in t, and there exists $T_p > 0$ such that $\forall\ (t, x, u) \in [0, \infty) \times \mathbb{R}^n \times \mathbb{R}^m$, $f(t + T_p, x, u) = f(t, x, u)$.

A-2 $f(t, 0, 0) = 0$ for all $t \geq 0$.

A-3 The user has selected an open ball about the origin, $B \subset \mathbb{R}^n$, a positive-definite, locally Lipschitz-continuous function $V : B \to \mathbb{R}$, and constants $0 < \alpha_1 \leq \alpha_2$ such that, $\forall\ x \in B$

$$\alpha_1 x^\top x \leq V(x) \leq \alpha_2 x^\top x.$$

$\blacksquare$

As alluded to above, the reader is encouraged to view the assumptions made throughout this section as requirements to impose on an open-loop trajectory optimizer. We have found them straightforward to meet when using the optimizers of [58, 53]. In many cases, the positive-definite function indicated in A-3 comes "for free" from the optimization problem used to compute trajectories; this is standard in model predictive control [76]. Because the Lyapunov condition in A-4 below can also be included as a constraint in the trajectory generation process, the user has much freedom in its selection, even something as simple as the 2-norm squared could be used.

6.1.2 Extracting a Feedback from Open-loop Trajectories

Two feedback controllers can be constructed from the following solutions of the model (6.5). The obvious relation to MPC is discussed in the remarks following Prop. VI.1.

A-4 There is a constant $0 \leq c < 1$, such that, for each initial condition $\xi \in B$, there exists a continuous input $u_\xi : [0, T_p] \to \mathbb{R}^m$ and a corresponding solution of the ODE, $\varphi_\xi : [0, T_p] \to \mathbb{R}^n$ satisfying $\varphi_\xi(T_p) \in B$, and

$$V(\varphi_\xi(T_p)) \leq cV(\xi); \tag{6.6}$$

moreover, for $\xi = 0$, $u_\xi(t) \equiv 0$. For clarity, solutions are taken in the sense of equation (C.2) of [63, pp. 657], namely

$$\varphi_\xi(t) = \xi + \int_0^t f(\tau, \varphi_\xi(\tau), u_\xi(\tau))d\tau. \tag{6.7}$$

■

A T_p-periodic Continuous-Hold (CH) feedback is defined by periodic extension of $u_\xi(t)$, namely,

$$u^{ch}(t, \xi) = u_\xi(\hat{t}),\ \hat{t} = t \bmod T_p. \tag{6.8}$$

Jumps are allowed at multiples of the period, with continuity taken from the right. Due to the reset or hold-nature of the above feedback, the stability of solutions of (6.5) in closed-loop with (6.8) should be studied as a sampled-data system, that is, the solutions should be evaluated at times $t_k = kT_p$. We will not analyze its stability, however, because it will clearly perform poorly in the face of perturbations occurring between samples, where the system is open loop.

We proceed directly instead to a feedback controller that allows continuous updates in the state variables, and yet, under certain conditions, can be built from the open-loop trajectories given in Assumption A-4. To understand the feedback controller, a *thought experiment* is helpful: Suppose at time $t_0 = 0$ the system's initial state value is ξ^0, and the continuous-hold feedback (6.8) is being applied. Then the system is evolving along the trajectory $\varphi_{\xi^0}(t)$. Suppose subsequently at time $0 < t_d < T_p$, an "impulsive disturbance" affects the system, displacing the system's state to a value $x(t_d) \neq \varphi_{\xi^0}(t_d)$. What input might be applied, given the information in A-4? If there exists a $\xi^d \in B$ such that $x(t_d) = \varphi_{\xi^d}(t_d)$, then applying the input $u_{\xi^d}(t)$ for $t_d \leq t < T_p$ will move the system toward the equilibrium in the sense that $V(\varphi_{\xi^d}(T_p)) \leq cV(\xi^d)$. The next result builds on this idea; see also [30, 108].

Proposition VI.1. *Assume the open-loop system (6.5) satisfies Assumptions A-1 to A-4. Assume in addition there exists an open set $B^e \supset B$ and a feedback*

$$\mu : [0, \infty) \times B^e \to \mathbb{R}^m$$

that is piecewise continuous in t, T_p-periodic, locally Lipschitz continuous in x, and, such that, for $0 \leq t < T_p$ and $\xi \in B$,

$$\mu(t, \varphi_\xi(t)) = u_\xi(t). \tag{6.9}$$

Then the origin of the closed-loop system,

$$\dot{x} = f^{cl}(t, x) := f(t, x, \mu(t, x)), \tag{6.10}$$

Features		Labels
t_j	$x^{j,i} = \varphi_{\xi^i}(t_j)$	$\mu^{j,i} = u_{\xi^i}(t_j)$
$t_0 = 0$	$x^{0,1}$	$\mu^{0,1}$
$t_0 = 0$	$x^{0,2}$	$\mu^{0,2}$
$\vdots$	$\vdots$	$\vdots$
$t_0 = 0$	$x^{0,M}$	$\mu^{0,M}$
t_1	$x^{1,1}$	$\mu^{1,1}$
$\vdots$	$\vdots$	$\vdots$
t_1	$x^{1,M}$	$\mu^{1,M}$
$\vdots$	$\vdots$	$\vdots$
$t_N = T_p$	$x^{N,1}$	$\mu^{N,1}$
$\vdots$	$\vdots$	$\vdots$
$t_N = T_p$	$x^{N,M}$	$\mu^{N,M}$

Table 6.1: Conceptual arrangement of the data in A-4 from which a controller satisfying (6.9) may be determined by "regression". In practice, not only time must be discretized, but also the initial conditions $\xi^i \in B$. This is where the "curse of dimensionality" rears its ugly head. Placing ten points per dimension leads to 10^n optimizations to compute, which quickly becomes impractical.

is locally exponentially stable, uniformly in t, and the trajectories in A-4 are solutions of (6.10)*. Said another way,* (6.10) *is a realization of the trajectories in A-4.* ■

Returning to (6.4) in the Introduction, the key point is that when a function can be found that is compatible with the open-loop trajectories in A-4 in the sense that the "learning condition " (6.9) is satisfied, then (6.10) provides an "exponentially stable realization" of the trajectories; this idea will be extended to a lower-dimensional system in the next subsection. Secondly, if at time $0 \leq t \leq T_p$ there exists ξ such that $x(t) = \varphi_\xi(t)$, the value of the "learned function" is being set to

$$\mu(t, x(t)) := u_\xi(t).$$

In practice, the trajectories will only be computed for a finite grid of initial conditions

ξ^j, $j \in J$. Hence, (6.9) is an *interpolation of the data* from the trajectory optimizations. For $x(t) \in B^e$ for which there does not exist ξ such that $x(t) = \varphi_\xi(t)$, the function in (6.9) is an *extrapolation of the data*. In [30], the fitting of a function to trajectory data was done *in principle* in closed form and by hand; here, the fitting is being done with Supervised Machine Learning. Moreover, the learning algorithms available today provide easy tools for checking the quality of a fit and hence for checking how closely a function was found that meets the learning condition (6.9).

Remark VI.2. Because the solutions in A-4 will be computed via a trajectory optimization algorithm, it is useful to understand how the assumptions on μ relate to requirements on the trajectories.

(i) Suppose $T_h > T_p$ and that for $\xi \in B$, $u^o_\xi : [0, T_h] \to \mathbb{R}^m$ minimizes a cost function of the form

$$J(\xi) = \min_u \int_0^{T_h} L(\varphi_\xi(t), u_\xi(t))dt + N(\varphi_\xi(T_h)) \tag{6.11}$$

where, as before, $\varphi_\xi(t)$ is the solution of (6.5) with initial condition ξ at $t_0 = 0$, and suppose furthermore that $u_\xi : [0, T_p] \to \mathbb{R}^m$ is the restriction of u^o_ξ to $[0, T_p]$, that is,

$$u_\xi = u^o_\xi\big|_{[0,T_p]}.$$

By the principle of optimality, for $0 \le t_0 < T_p$,

$$u^o_\xi\big|_{[t_0,T_h]}$$

is a minimizer of

$$J(x_0) = \min \int_{t_0}^{T_h} L(\varphi_\xi(t), u_\xi(t))dt + N(\varphi_\xi(T_h)), \tag{6.12}$$

where, $\varphi_\xi(t)$ is the solution of (6.5) with initial condition $\varphi_\xi(t_0)$ at t_0. Hence, the condition (6.9) can be interpreted as arising from an MPC-style controller

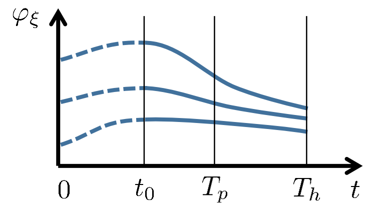


Figure 6.3: Principle of Optimality. If the system is initialized at $\varphi_\xi(t_0)$ and the cost function is modified from (6.11) to (6.12), then $\varphi_\xi : [t_0, T_h] \to \mathbb{R}^n$ is optimal.

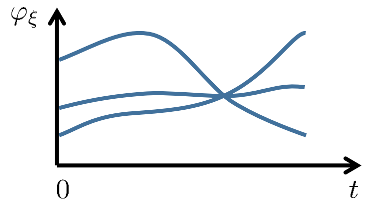


Figure 6.4: This shows the trajectories crossing one another, which means that the mapping $\Psi_t : B \to \mathbb{R}^n$ in (6.13) is not injective at certain moments of time. In this case, a feedback function cannot be extracted from the data.

with a shrinking horizon, $[t_0, T_h]$, for $0 \leq t_0 \leq T_p$ and fixed final-time T_h. This control strategy is visualized in Figure 6.3. The condition (6.9) comes from the shrinking of the optimization horizon; it will be essential in allowing a judiciously chosen set of open-loop trajectories to be realized with a low-dimensional state-variable model.

(ii) Supervised Machine Learning will be used to extract the function $\mu(t, x)$ in Prop. VI.1 from the trajectories and control inputs given in A-4 The method is sketched in Table 6.1. An example is given in Sect. 6.2.

(iii) The local Lipschitz continuity of μ imposes conditions on the solutions given in A-4. Indeed, for each $t \in [0, T_p]$, the mapping

$$\Psi_t : B \to \mathbb{R}^n, \text{ by } \Psi_t(\xi) := \varphi_\xi(t) \tag{6.13}$$

must be injective. This follows by the Gronwall-Bellman inequality [63, pp. 651]; see also [63, Exercise 3.17]. Hence, the optimization problem must be set up so as to avoid the existence of trajectories that cross one another, which can easily occur as shown in Figure 6.4. For example, if the user selected $T_h = T_p$ and imposed that the origin be attained at T_h, that is, dead-beat control, then a locally Lipschitz continuous μ would not exist.

(iv) Conditions are known under which the value function (6.11) meets the Lyapunov conditions in A-3 and A-4. Roughly speaking, they require that the terminal weight N either be replaced with finite-time convergence to the origin or, the terminal weight be selected as $\beta N(x(T_h))$, where $N(x)$ is positive definite and $\beta > 0$ is sufficiently large. Hence, in practice, the Lyapunov constraint can be replaced by careful formulation of the trajectory optimization problem. In our limited experience, it is never an active constraint.

(v) There is a long history of work in the nonlinear control literature that relates asymptotic controllability to an equilibrium point and the existence of stabilizing feedback controllers. The reader is referred to [30, 31, 25, 109] and references therein. The methods employed are not nearly as constructive as the work in this chapter.

6.1.3 Building a Reduced-Order Target Model

To begin the construction of a reduced-order target model as in (6.4), we now assume that the system (6.5) is decomposed in the form

$$\begin{aligned} \dot{x}_1 &= f_1(t, x_1, x_2) \\ \dot{x}_2 &= f_2(t, x_1, x_2, u), \end{aligned} \tag{6.14}$$

where $x_1 \in \mathbb{R}^{n_1}$ and $x_2 \in \mathbb{R}^{n_2}$. For clarity of exposition, the input is assumed not to appear in f_1; the changes required to include inputs in f_1 are given in Sect. 6.1.5. Assumptions A-1 through A-3 are assumed to hold for (6.14).

Because of how the decomposition will arise in the case of bipeds, we think of the x_1-states as the “weakly actuated part” of the system and the x_2-states as the “strongly actuated part” of the system. With the model expressed in this form, it is clear that the x_2-states are virtual controls for the x_1-states. We will continue to build open-loop trajectories by the full-order model, except now the trajectories will be computed for a *reduced set of initial conditions defined by the x_1-subsystem.*

Definition 1. *An insertion map, $\gamma : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$, is a function that preserves the equilibrium point, namely $\gamma(0) = 0$.* ■

The insertion map specifies initial conditions for x_2 as a function of x_1; in other words, it specifies the surface Z_0 in the Introduction, just before (6.2).

A-5 There is a constant $0 \leq c < 1$, such that, for each initial condition $\xi = (\xi_1, \gamma(\xi_1)) \in B$, there exists a piecewise continuous input $u_{\xi_1} : [0, T_p] \to \mathbb{R}^m$ and a corresponding solution of the ODE, $\varphi_{\xi_1} : [0, T_p] \to \mathbb{R}^n$ satisfying $\varphi_{\xi_1}(T_p) \in B$, and

$$V((\varphi_{1\xi_1}(T_p), \gamma(\varphi_{1\xi_1}(T_p))) \leq cV(\xi_1, \gamma(\xi_1)), \tag{6.15}$$

where the solution of the $(n_1 + n_2)$-dimensional model (6.14) has been decomposed as

$$\varphi_{\xi_1}(t) =: (\varphi_{1\xi_1}(t), \varphi_{2\xi_1}(t)).$$

■

Proposition VI.3. *Assume the open-loop system (6.14) satisfies Assumptions A-1 to A-3 and A-5, and define $B_1 := \{\xi_1 \in \mathbb{R}^{n_1} \mid (\xi_1, \gamma(\xi_1)) \in B\}$. Assume in addition there exists an open set $B_1^e \supset B_1$ and a function*

$$\nu : [0, \infty) \times B_1^e \to \mathbb{R}^{n_2}$$

that is piecewise continuous in t, T_p-periodic, locally Lipschitz continuous in x_1, and, such that, for $0 \leq t < T_p$ and $\xi_1 \in B_1$,

$$\nu(t, \varphi_{1\xi_1}(t)) = \varphi_{2\xi_1}(t). \tag{6.16}$$

Then the origin of the reduced-order system

$$\dot{x}_1 = f^{cl}_{\text{red}}(t, x_1) := f_1(t, x_1, \nu(t, x_1)), \tag{6.17}$$

is locally uniformly exponentially stable, and the trajectories in A-5 are solutions of (6.17). ■

Remark VI.4.

(i) Assumption A-5 and Prop. VI.3 represent our first result to mitigate the curse of dimensionality. Assumption A-5 leads to a greatly reduced training set for building a realization than A-4 because, in many practical examples, $n_1 \ll (n_1 + n_2)$. Proposition VI.3 says that this reduced training set can encode a stabilization goal that is a feasible action of the full-order model. The next section embeds the target model (6.17) in the full-order model, completing our basic plan for mitigating the curse of dimensionality.

(ii) The numerical burden of developing the "training sets" for $\nu(t, x_1)$ is exponential in the dimension of x_1, at least if a uniform grid is used to sample B_1.

(iii) Table 6.2 shows how to extract the function $\nu(t, x_1)$ from the optimization data.

(iv) The local Lipschitz continuity of $\nu(t, x_1)$ imposes stronger conditions on the solutions given in A-5 than those encountered in A-4. This is because the mapping defined by, for each $t \in [0, T_p]$,

$$\Psi_{1t} : B_1 \to \mathbb{R}^{n_1}, \ \ \Psi_{1t}(\xi_1) := \varphi_{1\xi_1}(t) \tag{6.18}$$

being injective is stronger than the mapping

$$\Psi_t : B_1 \to \mathbb{R}^n,\ \ \Psi_t(\xi_1) := \begin{bmatrix} \varphi_{1\xi_1}(t) \\ \varphi_{2\xi_1}(t) \end{bmatrix} \tag{6.19}$$

being injective. If Ψ_t is continuously differentiable and full rank, then there does exist a new choice of x_1-coordinates for which the corresponding mapping Ψ_{1t} is full rank and hence is locally injective. This will be illustrated on the cart-pendulum model in Sect. 6.2.

(v) Under the assumptions of Prop. VI.3, for each $t \in [0, T_p)$, $\Psi_t : B_1 \to \mathbb{R}^n$ is a homeomorphism onto its image. It follows that $\Psi_e : [0, T_p) \times B_1 \to \mathbb{R}^{n+1}$, by

$$\Psi_e(t, \xi_1) := \begin{bmatrix} t \\ \Psi_t(\xi_1) \end{bmatrix},$$

is also a homeomorphism onto its image. After augmenting the state with time in the usual manner, the low-dimensional model (6.4) discussed in the Introduction can be seen as evolving on the surface

$$Z_{T_p} := \Psi_e([0, T_p) \times B_1), \tag{6.20}$$

with the dynamics and output given by

$$\begin{aligned} \dot{\tau} &= 1 \\ \dot{x}_1 &= f_1(\tau, x_1, \nu(\tau, x_1)) \\ x &= \begin{bmatrix} x_1 \\ \nu(\tau, x_1) \end{bmatrix}. \end{aligned} \tag{6.21}$$

At this point, the direct relation with trajectories of the original model is only true for $0 \leq t < T_p$.

(vi) In Sect. 6.1.6, we will provide a concrete way to select the insertion map. For now, we propose an insertion map inspired by backstepping

$$\gamma(x_1) := K x_1, \tag{6.22}$$

that the equilibrium of the reduced-order model,

$$\dot{x}_1 = f_1(t, x_1, \gamma(x_1)) \tag{6.23}$$

is stable. Other relations to backstepping are noted in Appendix D.

(vii) Suppose the system (6.14) is time invariant, so that one is stabilizing a trivial periodic orbit (i.e., an equilibrium). Then $T_p > 0$ is a free parameter available to the designer. How to choose it? If the insertion map actually stabilizes the equilibrium of the reduced-order model (6.23), then in principle, T_p can be taken to be arbitrarily small, subject to choosing $c > 0$ and the positive function in (6.15) properly. Otherwise, if the system is locally asymptotically controllable to the origin [25], a larger T_p makes it easier to meet the Lyapunov contraction condition.

6.1.4 Embedding the Target Dynamics in the Original System

Consider the system (6.14) with the assumptions and notation of Prop. VI.3. Assume there exists a feedback $u(t, x_1, x_2)$ such that in the coordinates

$$y := x_2 - \nu(t, x_1), \tag{6.24}$$

Features		**Labels**
t_j	$x_1^{j,i} = \varphi_{1\xi_1^i}(t_j)$	$\nu^{j,i} = \varphi_{2\xi_1^i}(t_j)$
$t_0 = 0$	$x_1^{0,1}$	$\nu^{0,1}$
$t_0 = 0$	$x_1^{0,2}$	$\nu^{0,2}$
$\vdots$	$\vdots$	$\vdots$
$t_0 = 0$	$x_1^{0,M}$	$\nu^{0,M}$
t_1	$x_1^{1,1}$	$\nu^{1,1}$
$\vdots$	$\vdots$	$\vdots$
t_1	$x_1^{1,M}$	$\nu^{1,M}$
$\vdots$	$\vdots$	$\vdots$
$t_N = T_p$	$x_1^{N,1}$	$\nu^{N,1}$
$\vdots$		$\vdots$
$t_N = T_p$	$x_1^{N,M}$	$\nu^{N,M}$

Table 6.2: Conceptual arrangement of the data in A-5 from which a controller satisfying (6.16) may be determined by "regression". Since only the x_1-component of the state is sampled, far fewer optimizations are required. The number of time samples remains the same.

the closed-loop system has the form

$$\begin{aligned}\dot{x}_1 &= f_1(t, x_1, \nu(t, x_1) + y) \\ \dot{y} &= Ay,\end{aligned} \tag{6.25}$$

with A Hurwitz. Then the surface $y \equiv 0$ is invariant and the restriction dynamics is given by (6.17). While $\nu(t, x_1)$ is T_p-periodic, its limits from the left and the right are not necessarily equal at T_p, and y in (6.24) inherits this property. Hence, without further assumptions, it cannot be a solution of the ODE (6.25), in the usual sense [63, pp. 657], namely

$$y(t) = y(t_0) + \int_{t_0}^{t} Ay(\tau) d\tau.$$

In the following, we impose continuity in t on $\nu(t, x_1)$, and then after the theorem, analyze what this means in terms of the trajectories coming out of the optimizer.

Theorem VI.5. *Assume the open-loop system (6.14) satisfies Assumptions A-1 to A-3, and A-5, and define $B_1 := \{\xi_1 \in \mathbb{R}^{n_1} \mid (\xi_1, \gamma(\xi_1)) \in B\}$. Assume in addition there exists an open set $B_1^e \supset B_1$ and a feedback*

$$\nu : [0, \infty) \times B_1^e \to \mathbb{R}^{n_2} \tag{6.26}$$

that is continuous in t, T_p-periodic, locally Lipschitz continuous in x_1, and, such that, for $0 \le t < T_p$ and $\xi_1 \in B_1$,

$$\nu(t, \varphi_{1\xi_1}(t)) = \varphi_{2\xi_1}(t). \tag{6.27}$$

Then any feedback $u(t, x_1, x_2)$, piecewise continuous in t and locally Lipschitz continuous in x_1 and x_2 that transforms the system to (6.25), with A Hurwtiz, renders the origin of (6.25) locally uniformly exponentially stable. Moreover, the surface defined *by*

$$Z_e := \{(t, x_1, x_2) \mid (t \bmod T_p, x_1, x_2) \in Z_{T_p}\}, \tag{6.28}$$

is invariant with restriction dynamics given by (6.21). ■

Remark VI.6.

(i) In fact, (6.28) is the Isidori-Byrnes [57] Zero Dynamics Manifold and the Zero Dynamics is given by

$$\dot{\tau} = 1$$
$$\dot{x}_1 = f_1(\tau, x_1, \nu(\tau, x_1)).$$

The output that is being "zeroed" is

$$y = x_2 - \nu(t, x_1)$$

as long as the domain is properly specified.

(ii) If ν in (6.26) is continuous in t, then for all $x_1 \in B_1^e$, $\nu(T_p, x_1) = \nu(0, x_1)$. How does this relate to the trajectories in the training set used to generate ν? Because V in (6.15) is positive definite, and V decreases after T_p seconds, there exists an open ball B_2 contained in B_1 such that $\xi_1 \in B_2 \Rightarrow \varphi_{1\xi_1}(T_p) \in B_1$. Because $\hat{\xi}_1 := \varphi_{1\xi_1}(T_p) \in B_1$,

$$\varphi_{2\hat{\xi}_1}(0) = \gamma(\hat{\xi}_1) := \gamma(\varphi_{1\xi_1}(T_p)). \tag{6.29}$$

From (6.16), because $\hat{\xi}_1 \in B_1$,

$$\nu(0, \varphi_{1\hat{\xi}_1}(0)) = \varphi_{2\hat{\xi}_1}(0), \tag{6.30}$$

and because $\varphi_{1\hat{\xi}_1}(0) = \hat{\xi}_1$, we have

$$\nu(0, \varphi_{1\hat{\xi}_1}(0)) = \nu(0, \hat{\xi}_1). \tag{6.31}$$

From the continuity of ν and using the definition of $\hat{\xi}_1$,

$$\nu(0, \varphi_{1\xi_1}(T_p)) = \nu(T_p, \varphi_{1\xi_1}(T_p)). \tag{6.32}$$

From (6.16) again,

$$\nu(T_p, \varphi_{1\xi_1}(T_p)) = \varphi_{2\xi_1}(T_p). \tag{6.33}$$

Putting these together, the corresponding condition on the trajectories used in the training set for ν is given in A-6.

A-6 The solutions in A-5 also satisfy

$$\gamma(\varphi_{1\xi_1}(T_p)) = \varphi_{2\xi_1}(T_p). \tag{6.34}$$

■

Section 6.2 will illustrate these ideas on a simple low-dimensional example to make it easy for the interested reader to reproduce the results. The true benefits of the approach will not be clear until Sect. 6.4, where it will be applied to a high-dimensional hybrid model of a bipedal robot, and subsequently implemented in hardware on the robot MARLO.

6.1.5 Extended Class of Models

We discuss the case with the input appearing in both blocks. To keep the presentation brief and simple, it is supposed that the model has the form

$$\begin{aligned} \dot{x}_1 &= f_1(t, x_1, x_2, u) \\ \dot{x}_2 &= f_2(t, x_1, x_2, u), \end{aligned} \tag{6.35}$$

with

$$x_2 = \begin{bmatrix} x_{2a} \\ x_{2b} \end{bmatrix} \text{ and } f_2 = \begin{bmatrix} x_{2a} \\ u \end{bmatrix}.$$

Corollary VI.7. *Assume the open-loop system (6.35) satisfies Assumptions A-1 to A-3, A-5, and A-6, and define $B_1 := \{\xi_1 \in \mathbb{R}^{n_1} \mid (\xi_1, \gamma(\xi_1)) \in B\}$. Assume in addition there exists an open set $B_1^e \supset B_1$, a function*

$$\nu : [0, \infty) \times B_1^e \to \mathbb{R}^{n_2} \tag{6.36}$$

satisfying the conditions of Theorem VI.5, and a second function

$$\mu : [0, \infty) \times B_1^e \to \mathbb{R}^{m} \tag{6.37}$$

that is piecewise continuous in t, T_p-periodic, locally Lipschitz continuous in x_1, and such that, for $0 \leq t < T_p$ and $\xi_1 \in B_1$,

$$\mu(t, \varphi_{1\xi_1}(t)) = u_{\xi 1}(t). \tag{6.38}$$

Then for all ${}^{n_2}/_2 \times {}^{n_2}/_2$ positive definite matrices K_p and K_d, the origin of

$$\begin{aligned} \dot{x}_1 &= f_1(t, x_1, x_2, u) \\ \dot{x}_2 &= f_2(t, x_1, x_2, u) \\ u &= \mu(t, x_1) - [K_p\ K_d]\big(x_2 - \nu(t, x_1)\big) \end{aligned} \tag{6.39}$$

is locally exponentially stable, uniformly in t_0. Moreover, the surface defined by (6.28) is invariant with restriction dynamics given by

$$\begin{aligned} \dot{\tau} &= 1 \\ \dot{x}_1 &= f_1(t, x_1, \nu(t, x_1), \mu(t, x_1)) \\ x &= \begin{bmatrix} x_1 \\ \nu(\tau, x_1) \end{bmatrix}. \end{aligned} \tag{6.40}$$

■

Remark VI.8.

(i) There is no extra boundary condition, such as A-6, associated with (6.27) because the term μ arises from the inputs instead of the states of the ODE, as in the case of ν. In particular, μ can be piecewise continuous in t. The learning of μ is done the same as for ν in Table 6.2.

(ii) Most systems will require a pre-feedback to arrive at the form (6.35); this must be taken into account when implementing the feedback indicated in (6.39).

6.1.6 Orbit Library and Design of the Insertion Map

The objective of this section is to provide a systematic means for designing the insertion map in a way that takes into account the "physics" of a model.

Definition 2. *An orbit library $\mathcal{L}$ is a set of periodic trajectories of the model* (6.35) *that are parameterized by the x_1-states. We denote the library consisting of the periodic solutions by*

$$\mathcal{L} := \{\varphi_{\xi_1} : [0, T_p] \to \mathbb{R}^n \mid \xi_1 \in B_1\}, \tag{6.41}$$

with B_1 as A-5. ■

Definition 3. *An insertion map associated to an orbit library* (6.41) *is a function $\gamma_{\mathcal{L}} : B_1 \to \mathbb{R}^{n_2}$ such that*

$$\gamma_{\mathcal{L}}(\xi_1) := \varphi_{2\xi_1}(0). \tag{6.42}$$

■

One should think of the above insertion map as taking the states of the x_1-coordinates, associating them to periodic orbits of the full model, and then defining the initial condition of the x_2-coordinates (in the trajectory optimization) to be its value at a point on the associated periodic orbit. Hence, the overall model is being initialized in a physically meaningful manner. Moreover, the trajectories in A-6 can now be interpreted as affecting a transition *from* a family of periodic solutions *to* a desired periodic solution *in a way that* leads to stabilization of the desired periodic solution, via Theorem VI.5 or Corollary VI.7. The authors have found this to be very useful on bipedal robots.

6.2 Inverted Pendulum on a Cart

This section will illustrate the controller designs of Section 6.1 on the well-known inverted pendulum on a cart model. The MATLAB code for the calculations is available for download in Extension 1. The optimization setup and the learning method used here are nearly identical to what will be implemented on a bipedal robot in Section 6.4; the only significant change involves the hybrid aspect of a biped model.

6.2.1 System Model

The system consists of a unit length, uniformly distributed unit-mass pendulum attached via a revolute joint to a planar unit-mass cart, shown in Figure 6.5. A driving force is applied on the cart, and there is no torque acting on the revolute joint of the pendulum. The motion of the cart and the pendulum are free of friction forces. The configuration variable $q := (p, \theta)$ consists of the cart position and the pendulum angle. The system is written in state variable form as

$$\begin{aligned} \dot{x}_1 &= \begin{bmatrix} \dot{p} \\ \frac{2\sin(\theta)\dot{\theta}^2 - 3g\cos(\theta)\sin(\theta) - 4u}{3\cos(\theta)^2 - 8} \end{bmatrix} \\ \dot{x}_2 &= \begin{bmatrix} \dot{\theta} \\ \frac{3\cos(\theta)\sin(\theta)\dot{\theta}^2 - 12g\sin(\theta) - 6\cos(\theta)u}{3\cos(\theta)^2 - 8} \end{bmatrix}, \end{aligned} \tag{6.43}$$

where u is the force acting on the cart and the system state x is decomposed into $x_1 = (p, \dot{p})$ and $x_2 = (\theta, \dot{\theta})$.The equilibrium point of the upright pendulum is $x^* = 0$ and $u^* = 0$. Assumptions A-1 and A-2 are then trivially satisfied.

The overall control objective will be to locally exponentially stabilize a continuum of periodic motions with a common period $T_p = 2$ seconds. We first illustrate the control design method on a trivial periodic orbit corresponding to the pendulum upright and the cart at the origin.

6.2.2 Stabilizing the Upright Equilibrium While Respecting a Barrier

The presentation follows the basic steps of the design, from learning a full-state feedback as in Prop. VI.1 to embedding a target model as in Corollary VI.7.

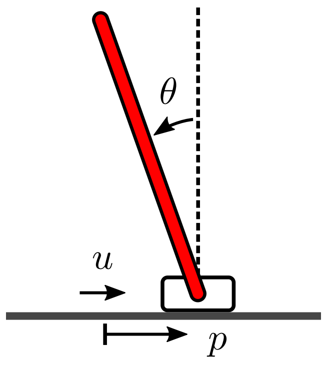


Figure 6.5: An inverted pendulum on a cart model is used to illustrate the controller designs of Sect. 6.1. The objective is to stabilize a continuum of periodic motions, including a trivial periodic orbit corresponding to the pendulum upright and the cart at the origin. In part of the analysis, a barrier is imposed.

6.2.2.1 Trajectory Generation and Learning for the Full-Order Model

The set B and positive definite function of A-3 are discussed shortly. For an initial state $\xi \in B$, the direct collocation algorithms of [58, 53] are used to generate a trajectory $\varphi_\xi(t)$ and corresponding input $u_\xi(t)$ over an interval $[0, T_p]$ to meet the conditions of A-4. To emphasize the ability to handle interesting constraints in the control design, the cart position is heavily penalized if it moves out of a "safe region" $[-p_b, p_b]$, with $p_b = 2$.

The cost function for determining the trajectories is a standard quadratic form with an additional penalty for the safety region:

$$\begin{aligned} J(\xi) &= \min_u \int_0^{T_h} \big(||x||_Q^2 + ||u||_R^2 + L(p, p_b)\big) dt \\ L(p, p_b) &= wp^2(e^{p-p_b} + e^{-p-p_b}). \end{aligned} \tag{6.44}$$

The weights Q and R are taken as identity matrices and the penalty weight is $w = 10$. The optimization is subjected to the system dynamics constraints(6.43) and the terminal constraint $x(T_h) = 0$, with $T_h = 3T_p = 6$ seconds. One could also use a terminal cost $N(x(T_h))$ in place of the terminal constraint. Even though a terminal constraint may make the optimization problem infeasible for some initial conditions,

Table 6.3: MATLAB Neural Network Fitting Parameters

Hidden neurons	50
Training Ratio	80%
Validation Ratio	20%
Training Algorithm	Bayesian Regularization
Max Iteration	4000

we have found it to be quite practical in bipedal robots. As discussed earlier, the cost function in (6.44) can often used as a Lyapunov function meeting the conditions in A-4. Here, we do not add this as a constraint to the optimization and will illustrate the satisfaction of the Lyapunov condition.

The function $\mu(t, x)$ in Prop. VI.1 is learned for the ball of initial conditions

$$\begin{aligned} B =& \{-1 \leq p \leq 1, -\frac{\pi}{6} \leq \theta \leq \frac{\pi}{6}, \\ & -2 \leq \dot{p} \leq 2, -2 \leq \dot{\theta} \leq 2\}, \end{aligned} \tag{6.45}$$

with samples $\xi^i \in B$ selected from a uniform grid of the state. Five points are used in each dimension, for a total of 625 input sequence $u_{\xi^i}(t)$ and solutions $\varphi_{\xi^i}(t)$. At each time

$$t_j \in \{t \mid j\frac{T_p}{40}, j = 0, 1, \dots, 40\}, \tag{6.46}$$

the time-state pair $(t_j, x^{j,i})$ is a feature and the input $u^{j,i}$ is a label. The complete list of features and labels is shown in Table. 6.1 in Section 6.1.2. We use the MATLAB Neural Network Fitting Toolbox to approximate μ. The fitting setup is shown in Table 6.3. The mean squared error of the validation set is around 10^{-4}.

We show there exists a Lyapunov function

$$V(x) := x^\top P x \tag{6.47}$$

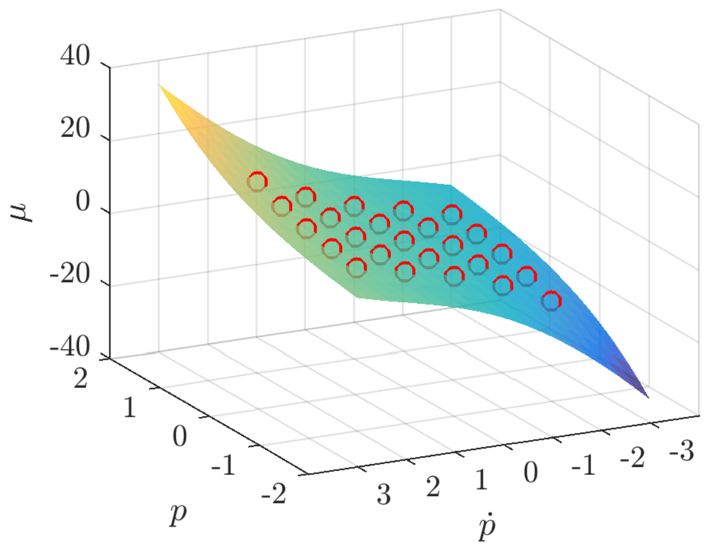


Figure 6.6: A slice of the function $\mu(t, p, \dot{p}, \theta, \dot{\theta})$, with $t = 0$, $\theta = 0$ and $\dot{\theta} = 0$. The presence of the (soft-penalty) barrier is most evident near $p = -2$ and $\dot{p} = -3$. The circles are training and validation data. Both interpolation and extrapolation can be seen in the surface.

as required in A-4 that is built from the cost function in (6.44). The matrix

$$P = \begin{bmatrix} 0.04 & -0.11 & 0.03 & -0.03 \\ -0.11 & 0.94 & -0.12 & 0.18 \\ 0.03 & -0.12 & 0.03 & -0.03 \\ -0.03 & 0.18 & -0.03 & 0.04 \end{bmatrix} \tag{6.48}$$

is from a regression of $J(x)$. The matrix is positive definite. We next find the constant c in A-4 using the data set to train $\mu(t, x)$ that c satisfies

$$c \geq \max_i \frac{V(\varphi_{\xi^i}(T_p))}{V(\varphi_{\xi^i}(0))}. \tag{6.49}$$

The maximum ratio over 625 points of ξ^i is 0.22, then we set $c = 0.25$. Notice when ξ^i is the equilibrium point, $V(\varphi_{\xi^i}(T_p) = V(\varphi_{\xi^i}(0)) = 0$, which has to be ignored when finding c. Figure 6.7 shows that $V(\varphi_\xi(kT_p))$ in the simulation is exponentially decreased.

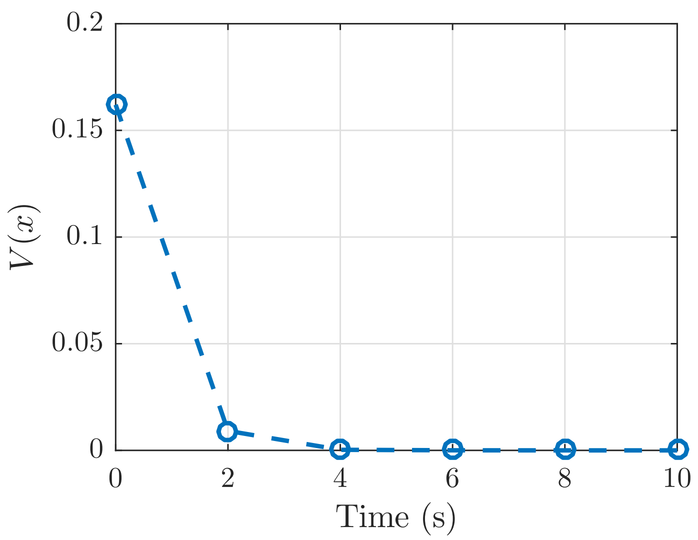


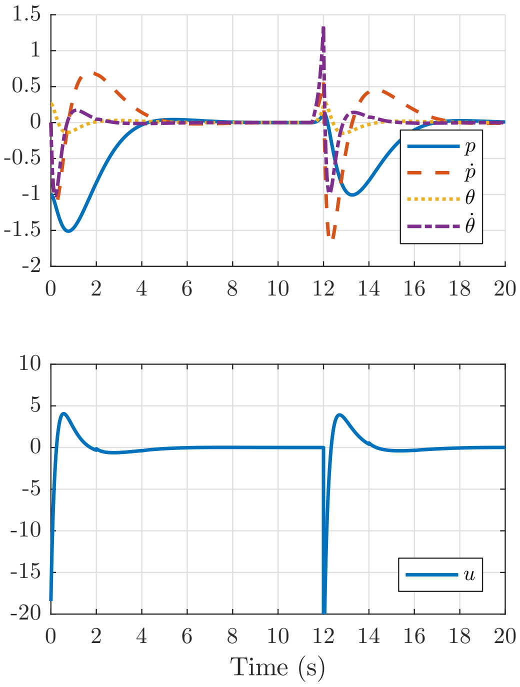
Figure 6.7: The plot shows the typical evolution of the optimization-cost function, confirming that it serves as a Lyapunov function. It is to be noted that $V(t)$ is only required to monotonically decrease from sample to sample, that is, from kT_p to $(k+1)T_p$, with $T_p = 2$.

6.2.2.2 Comparison of Continuous Hold vs Learned Feedback

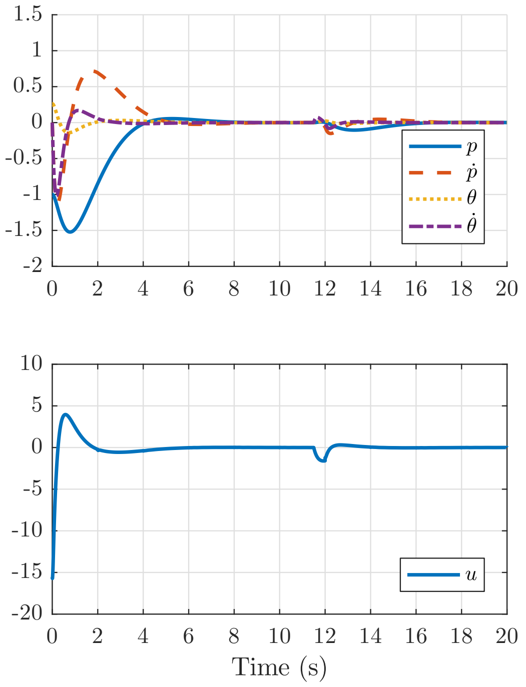
Figure 6.8 compares u^{ch} in (6.8) and μ in (6.9). The continuous-hold feedback u^{ch} updates the state every $T_p = 2$ seconds. This type of MPC-style controller is guaranteed to perform poorly in the face of disturbances occurring within the sample period. Figure 6.8-(a) shows the continuous-hold controller stabilizing the system from the initial condition

$$(p, \dot{p}, \theta, \dot{\theta}) = (-1, 0, {}^{\pi}/_{12}, 0). \tag{6.50}$$

Figure 6.8-(b) shows that the learned feedback μ performs identically to u^{ch} given the same initial condition and a perfect model. The difference is obvious when it comes to disturbance rejection. A constant external force $d = 1N$ is applied to the cart for $t \in [11.5, 12]$. The continuous-hold feedback u^{ch} has to wait till $t = 12$ to update the state and respond to the disturbance; on the other hand, the learned feedback μ updates the state continuously and hence reacts to the disturbance immediately.



(a) Continuous-hold controller u^{ch}



(b) Learned feedback μ

Figure 6.8: Comparison of the continuous-hold controller in (a), versus the learned controller in (b). In both cases, the initial condition is $(p, \dot{p}, \theta, \dot{\theta}) = (-1, 0, {}^{\pi}/_{12}, 0)$. A disturbance is applied for $t \in [11.5, 12]$ seconds. The classical MPC plot is in an Appendix-D

6.2.2.3 Building a Reduced-Order Target Model

The previous subsection illustrated how feedback is extracted from data via Supervised Machine Learning and will provide a benchmark for later designs. Because the cart-pendulum system has only four states, the number of trajectories required for training in the previous subsection was quite manageable, and with random sampling techniques [85], it could be further reduced. Eventually, however, the number of required optimizations will become untenable. Here we illustrate Prop. VI.3 for a two-dimensional subsystem of the cart-pendulum system.

Recall that the system state decomposition was already shown in (6.43). The insertion map used here is inspired by backstepping as in Remark VI.4. Linearizing the x_1-subsystem with $u = 0$ and selecting x_2 as a stabilizing linear feedback yields

$$\gamma(x_1) = \begin{bmatrix} 0.03 & 0.1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \end{bmatrix}. \tag{6.51}$$

We do not further explore the choice of γ because we will primarily use the orbit library $\gamma_{\mathcal{L}}$ of Definition 3 in the remainder of the chapter, including the bipedal robot section.

With this insertion map, the trajectories required by A-5 are determined via optimization with

$$B_1 = \{-1 \leq p \leq 1, -2 \leq \dot{p} \leq 2\}. \tag{6.52}$$

In anticipation of using the results here in Corollary VI.7, the boundary condition of A-6 is also imposed.

The set of initial conditions $\xi_1^j,\ j \in J$ now has 25 points instead of 625 points. The mapping (6.18) is checked to be injective by evaluating the numerical rank of the x_1-features in Table 6.2 via SVD. Just as before, the function $\nu(t, x_1)$ is obtained from the data via the MATLAB Neural Network Fitting Toolbox, again with the

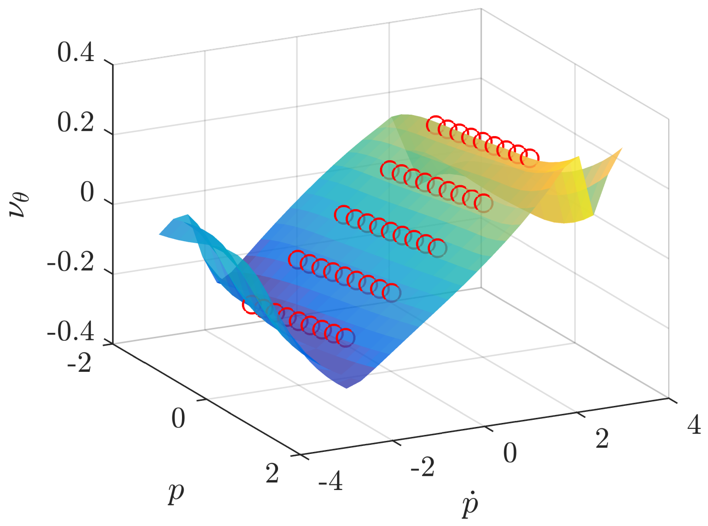


Figure 6.9: A slice of the function $\nu(t, p, \dot{p})$, with $t = 0$. This is the ν associated with x_2 coordinate θ. The circles are training and validation data. The interpolation is smooth while the extrapolation may be not.

parameters indicated in Table 6.3. The same holds for the function $\mu(t, x_1)$. An example of the fitting of ν is shown in Figure 6.9.

The evolution of the target model is shown in the next subsection when a disturbance is applied after $y = x_2 - \nu(t, x_1)$ has nearly converged to zero.

6.2.2.4 Embedding the Target Dynamics in the Original System

The learned functions from the reduce-order optimization are now used to stabilize the full-order system based on Theorem VI.5 and Corollary VI.7. To place the system in the form (6.35), a pre-feedback is applied

$$\bar{u} := \frac{3\cos(\theta)\sin(\theta)\dot{\theta}^2 - 12g\sin(\theta) - 6\cos(\theta)u}{3\cos(\theta)^2 - 8} \tag{6.53}$$

resulting in

$$\ddot{\theta} = \bar{u}.$$

The original input u can be computed from $\bar{u}$ because (6.53) is invertible in the operational range of interest, namely $-{}^{\pi}/_{2} < \theta < {}^{\pi}/_{2}$. While the function $\bar{\mu}(t, x)$ of

Corollary VI.7 can be recovered from $\mu(t,x_1)$ and (6.53), it is just as easy to learn it with the features $(t_j, x_1^{j,i})$ and label $\bar{u}^{j,i}$. In the full model,

$$\bar{u} = \bar{\mu}(t,x_1) - [K_p \; K_d]\big(x_2 - \nu(t,x_1)\big), \tag{6.54}$$

with $K_p = 50$ and $K_d = 15$.

Figure. 6.10 shows the response of the closed-loop system with the same initial condition and perturbation of Figure. 6.8. The settling time and disturbance rejection performance is similar to the full state learned feedback. Figure 6.11 illustrates the attractiveness of the surface $x_2 = \nu(t,x_1)$ by showing that the output error in (6.24) of the full-order system decays exponentially to zero. When the disturbance is applied for $11.5 \le t < 12$, the output is driven away from zero and then decays back quickly when the disturbance is removed.

6.2.3 Orbit Library and Transitioning Among Periodic Orbits

The last subsection has gone through the control design process for a trivial periodic orbit where the pendulum is upright, and the cart is at the origin. This subsection designs a controller for a set of periodic orbits, illustrate an insertion map $\gamma_{\mathcal{L}}$ arising from an orbit library, and shows the possibility of the mapping (6.18) not being injective. To simplify matters, we work directly with the cart-pendulum system *after* the pre-feedback (6.53) has been applied.

6.2.3.1 Orbit Library

For $T_p = 2$ seconds, define a set of periodic motions of the cart by

$$p(t) = p_0 + \frac{\dot{p}_0}{\pi}\sin(\pi t),$$

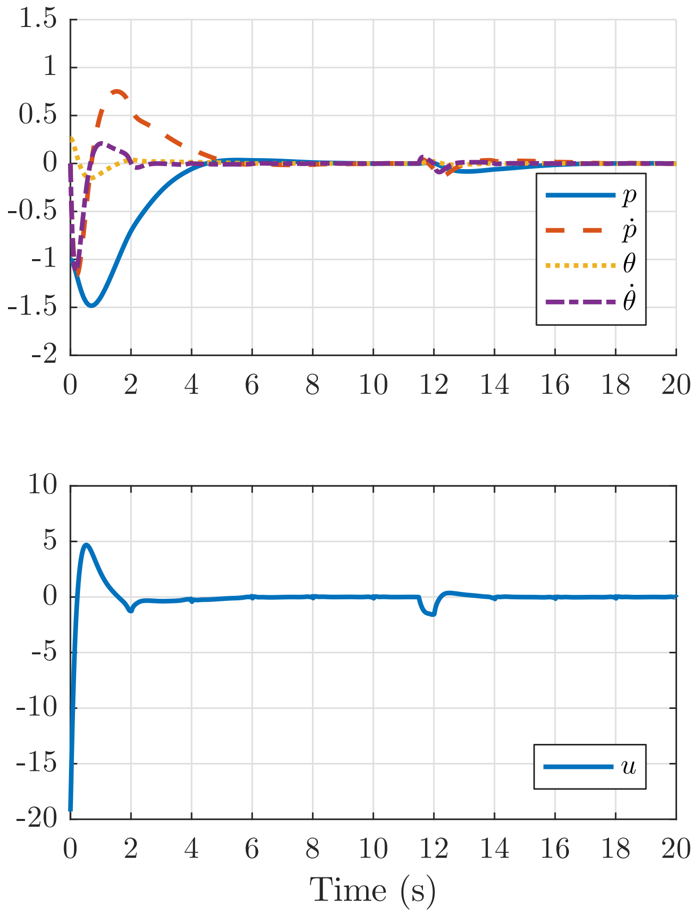


Figure 6.10: Response of the reduced-order model (6.40). The states of the model are p and $\dot{p}$, while θ and $\dot{\theta}$ are outputs. The initial condition and disturbance are as in Figure 6.8.

for $(p_0, \dot{p}_0) \in B_1$ in (6.52). The trajectory for $p(t)$ fixes the acceleration of the cart, which in turn gives trajectories for $\theta(t)$, $\dot{\theta}(t)$, and $u(t)$. Moreover, imposing $-{}^{\pi}/_{2} < \theta < {}^{\pi}/_{2}$ selects among the two possible solutions for the model. These considerations define an orbit library $\mathcal{L}$, with solutions indexed by $(p_0, \dot{p}_0)$. Denote the set of initial conditions of the orbit library as $(\xi_1^{\mathcal{L}}, \xi_2^{\mathcal{L}})$. Recalling Definition 3, an insertion map associated to the orbit library is

$$\gamma_{\mathcal{L}}(\xi_1^{\mathcal{L}}) = \xi_2^{\mathcal{L}}.$$

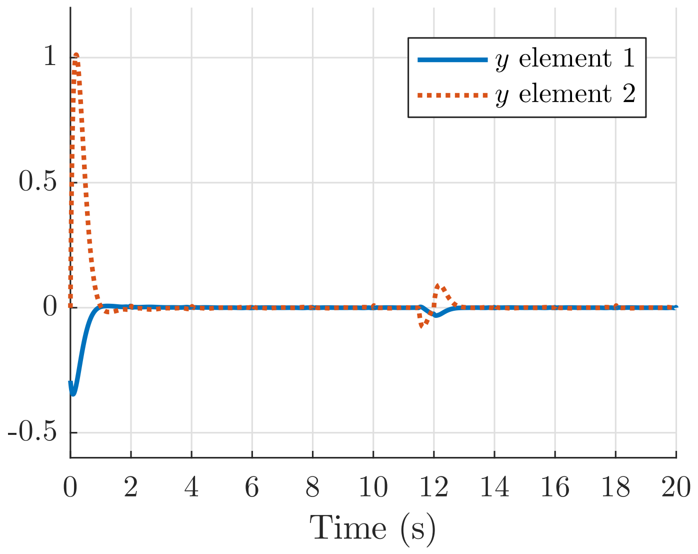


Figure 6.11: Showing the convergence of the output error in (6.24). Comparing this figure with Figure 6.10 shows that the system converges to the zero dynamics surface more quickly than it converges to the periodic orbit. The disturbance initially drives the system away from the surface.

To make it more explicit, for this example, we use linear regression to find $\gamma_{\mathcal{L}} : \mathbb{R}^2 \to \mathbb{R}^2$ as

$$\begin{bmatrix} \theta_0 \\ \dot{\theta}_0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0.5911 \end{bmatrix} \begin{bmatrix} p_0 \\ \dot{p}_0 \end{bmatrix}. \tag{6.55}$$

Remark VI.9. The reader may be wondering why we bring up the orbit library as a means of computing a new insertion function, especially when the 'backstepping-inspired' insertion map worked so well? The point is that for a robot, where the dimension of x_2 may be twenty, one has no idea how to design a 'backstepping-inspired' insertion map, whereas the concept of an orbit library extends naturally as will be seen later in the chapter.

6.2.3.2 Loss and Recovery of Injectivity

Using the new insertion function (6.55), trajectory generation is performed exactly as in Sect. 6.2.2, with x_1 and x_2 as given in (6.43). The mapping (6.18) is checked not to be injective. Indeed, for $t \simeq 1.8$, the cart trajectories pass through a one-

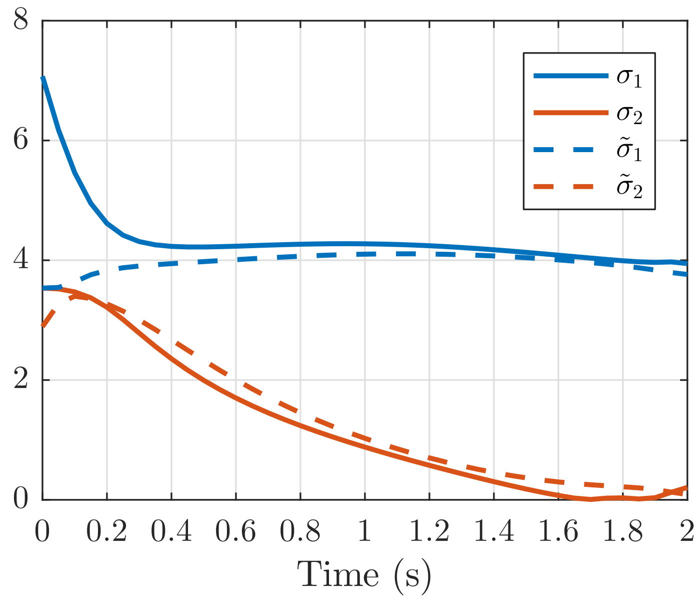


Figure 6.12: The singular values of the matrix formed by the sampled trajectories versus time the insertion function is given by (6.55). The solid lines are for $(p(t), \dot{p}(t))$, with $\Delta t = 0.05$, while the dashed lines are for $(p(t) - \theta(t), \dot{p}(t) - \dot{\theta}(t))$ for the same time samples. For the choice of the insertion function arising from backstepping in (6.51), the minimum value of σ_2 was 0.58.

dimensional surface as shown in Figures 6.12 and 6.13, while the mapping (6.19) remains injective. Hence, one expects the existence of a set of coordinates in which the design can proceed. It can be checked that the new coordinates[2]

$$\tilde{x}_1 = (p - \theta, \dot{p} - \dot{\theta})$$

are "full rank" as shown in Figure 6.12. It is not necessary to redo the optimization because the new coordinates correspond to a new $\tilde{B}_1$ and a new insertion map, $\tilde{\gamma}_{\mathcal{L}}$, and these are not explicitly required in the computation of ν and μ. The important thing is the feature set for the Supervised Machine Learning is now indexed by $(t_j, \tilde{x}_1^{j,i})$ rather than $(t_j, x_1^{j,i})$.

[2]Almost any linear combination of x_1 and x_2 works because it takes rows from the bottom of (6.19) and adds them to the top rows, making (6.18) full rank, and hence locally injective. There is nothing magic about our choice.

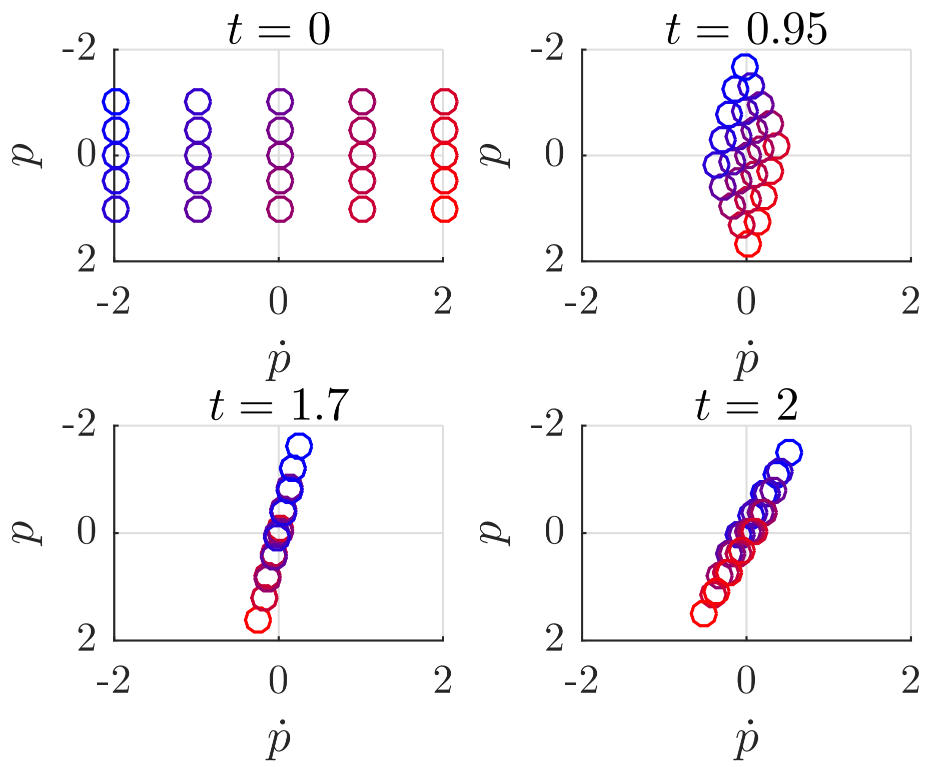


Figure 6.13: Another perspective on the information in Figure 6.12. The initial conditions are taken from a grid, as can be seen at $t = 0$. At subsequent times, the grid is transformed into a parallelogram at $t = 0.95$ and a line at $t = 1.7$, where the mapping $\Psi_{1t} : B_1 \to \mathbb{R}^{n_1}$ in (6.18) loses rank. We have not yet observed this problem in the case of bipedal robots.

6.2.3.3 Transitioning Between Periodic Orbits

Next, we use the library insertion map $\gamma_{\mathcal{L}}$ and the new state $\tilde{x}_1$ to design a controller for transitioning between periodic orbits; this is analogous to transitioning between walking gaits of various speeds or direction for a bipedal robot. The cost function used in A-5 is modified to include the target orbit $A := (p_0, \dot{p}_0) \in B_1$, per (6.56) to

$$J(\xi_1, A) = \min_u \int_0^{T_h} \left(||x - \varphi^A||_Q^2 + ||u - u^A||_R^2 \right) dt \tag{6.56}$$

subject to $x(T_h) = \varphi^A(T_p)$ and $x(0) = (\xi_1, \gamma_{\mathcal{L}}(\xi_1))$. The boundary condition A-6 is also applied as $\gamma_{\mathcal{L}}(x_1(T_p)) = x_2(T_p)$. Here, the target trajectory and its corresponding input are denoted as $\varphi^A(t)$ and $u^A(t)$. To simplify the problem, the cost function excludes the barrier penalty $L(p, p_b)$ in (6.44). The set B_1 is still given by (6.52).

With this choice of the insertion map, the boundary condition A-6 means that each cycle in the transition moves the cart-pendulum from one periodic orbit to the next; this is because $(\varphi_{1\xi_1}(T_p), \gamma_{\mathcal{L}}(\varphi_{1\xi_1}(T_p))$ is an initial condition for a periodic solution of the model. Denote the family of solutions to the optimization problem by

$$\begin{aligned} \tilde{x}_1^{i,j,k} &:= \varphi_{1\xi_1^i}^{A_k}(t_j) - \varphi_{2\xi_1^i}^{A_k}(t_j) \\ \nu^{i,j,k} &:= \varphi_{2\xi_1^i}^{A_k}(t_j) \\ \bar{\mu}^{i,j,k} &:= \bar{u}_{\xi^i}^{A_k}(t_j). \end{aligned} \tag{6.57}$$

The feature set for the Supervised Machine Learning is taken as $(t_j, \tilde{x}_1^{i,j,k}, A_k)$ and the labels are $(\nu^{i,j,k}, \bar{\mu}^{i,j,k})$. Figure 6.14 shows a orbit transition

$$(-1, 0.5) \to (0, 0) \to (0, 1.2) \tag{6.58}$$

of the target orbit $(p_0, \dot{p}_0)$ at $t = 20$ and $t = 40$. A constant external force $d = 20N$ is applied to the cart for $t \in [69.5, 70]$.

Remark VI.10.

(i) Orbit transition from set B_1 to a target orbit A can also be reviewed as rejecting state disturbances in B_1. The distance from a state in B_1 to A is not necessarily "small", indicating the region of attraction for this controller could be "large".

(ii) There may exist two orbits A_m and A_n in B_1 for which a transition cannot be achieved over $[0, T_h]$. However, one may think of transitions in B_1 as a graph so that if there exists an orbit A_k such that

$$A_m \to A_k \to A_n$$

is possible, then the orbits are connected.

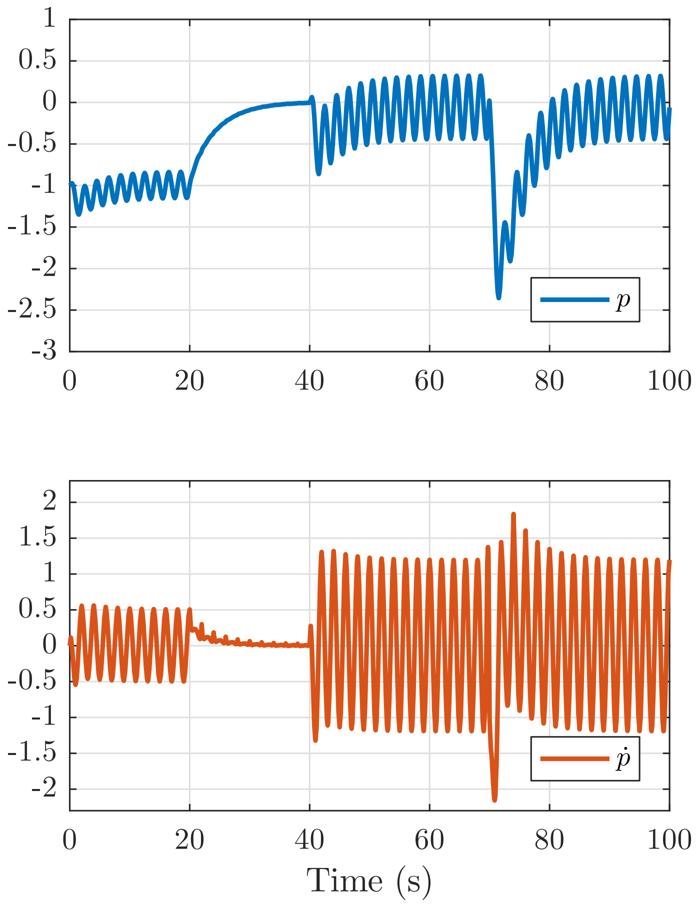


Figure 6.14: Plots of $p(t)$ (top) and $\dot{p}(t)$ (bottom) as the closed-loop system transitions from one periodic orbit to another, as given in (6.58), with a disturbance applied for $t \in [69.5, 70]$.

(iii) If the target orbit is modified at multiples of T_p, there are no jumps in ν; this is because the orbit-library insertion map transitions the system from one periodic orbit to another as shown in Figure 6.15.

6.3 Hybrid Model and Control

This section describes an extension of the control policy developed in Sect. 6.1 to systems with impulse effects [50, 125, 15], a special class of hybrid models that arises in bipedal robots. The control goals for the hybrid system corresponds to stabilizing periodic walking gaits for various speeds, and to transitioning among these gaits. The

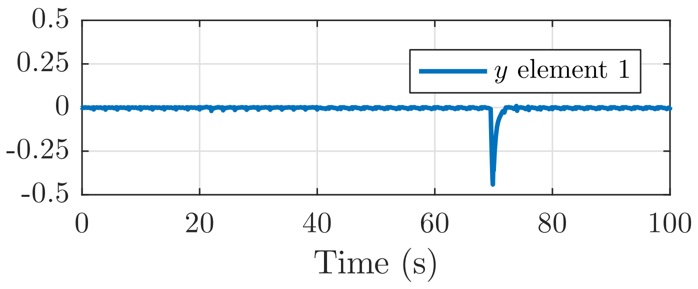


Figure 6.15: This shows that there is no jump in the output when the transition point takes place at a multiple of T_p. The jump corresponds to the disturbance in Figure 6.14. Only the first component of (6.24) is shown as the other component is the derivative of this one.

robot should also be able to reject a range of force perturbations.

6.3.1 Hybrid Model

We consider a hybrid system with one continuous-time phase as follows

$$\Sigma : \begin{cases} \dot{x} \;= f(x, u) & x^- \notin \mathcal{S} \\ x^+ = \Delta(x^-) & x^- \in \mathcal{S}, \end{cases} \tag{6.59}$$

in which $x \in \mathcal{X}$ and $\mathcal{X} \subset \mathbb{R}^n$ denote the *vector of state variables* and n-dimensional *state manifold*, respectively. The continuous-time control input is represented by $u \in \mathcal{U}$, where $\mathcal{U} \subset \mathbb{R}^m$ is an open *set of admissible control values*. In addition, $f : \mathcal{X} \times \mathcal{U} \rightarrow \mathrm{T}\mathcal{X}$ is assumed to be continuously differentiable ($\mathcal{C}^1$) so that a Poincaré map can be computed later when checking stability. For each $u \in \mathcal{U}$, $f(\cdot, u)$ is a vector field in $\mathrm{T}\mathcal{X}$, the *tangent bundle* of the state manifold $\mathcal{X}$.

The *switching hypersurface* $\mathcal{S}$ is an $(n-1)$-dimensional manifold

$$\mathcal{S} := \{x \in \mathcal{X} \,|\, p(x) = 0\}\,, \tag{6.60}$$

on which the state solutions are allowed to undergo a sudden jump according to the *re-initialization rule* $x^+ = \Delta(x^-)$. Here, $p : \mathcal{X} \rightarrow \mathbb{R}$ is a $\mathcal{C}^1$-*switching function* which satisfies $\frac{\partial p}{\partial x}(x) \neq 0$ for all $x \in \mathcal{S}$. Moreover, $\Delta : \mathcal{X} \to \mathcal{X}$ denotes the $\mathcal{C}^1$ *reset map*. $x^-(t) := \lim_{\tau \nearrow t} x(\tau)$ and $x^+(t) := \lim_{\tau \searrow t} x(\tau)$ represent the left and right limits of the state trajectory $x(t)$, respectively. As in [122], the solution of the hybrid system (6.59) is assumed to be right continuous. In particular, it is constructed by piecing together the flow of $\dot{x} = f(x, u)$ such that the discrete transition takes place when this flow intersects the switching hypersurface $\mathcal{S}$. The new initial condition for $\dot{x} = f(x, u)$ is then determined by the reset map $x^+ = \Delta(x^-)$.

6.3.2 Setting up the Optimization Problem

We now start the translation of the Assumptions A-1 to A-6 to the hybrid setting for the purpose of designing a feedback controller to locally exponentially stabilize a periodic solution. For bipedal robots, mid-step is a good time to make adjustments to the gait: (a) impact transients have had a chance to settle out; (b) the swing foot is safely away from the ground; and (c) there is still adequate time to steer the swing leg to a favorable configuration for impact. Hence, we will use mid-step of the periodic orbit understudy for setting the beginning and end of the trajectories that we compute via optimization.

The hybrid model (6.59) is assumed to satisfy the following conditions.

H-1 $f : \mathcal{X} \times \mathcal{U} \to T\mathcal{X}$ and the reset map $\Delta\mathcal{X} \to \mathcal{S}$ are Lipschitz continuous. This will allow the stability analysis tools of [4] to be applied later on.

H-2 There exists $T_p > 0$, $x^*_{\mathrm{m}} \in \mathbb{R}^n$, a piecewise continuous input $u^* : [0, T_p] \to \mathcal{U}$ and a solution $\varphi^*(t)$ of (6.59) satisfying:

- **a)** $\varphi^{*+}(0) = x^*_{\mathrm{m}}$;
- **b)** $\varphi^{*-}({}^{T_p}\!/_2) \in \mathcal{S}$, (swing foot touches the ground);
- **c)** $\forall\, t \neq {}^{T_p}\!/_2, \varphi^*(t) \notin \mathcal{S}$, (does so only once); and
- **d)** $\varphi^{*-}(T_p) = x^*_{\mathrm{m}}$ (periodicity).

It is noted that by the definition of $\mathcal{S}$, the periodic solution is transversal to $\mathcal{S}$, namely $\frac{d}{dt}p(\varphi^{*-}(T_p/2)) < 0$. And yes, the motion is being "clocked" with the middle of the step.

■

The point $x^*_{\rm m}$ is the midpoint of the periodic trajectory, as measured by time. The controller we build will start from mid-stance, follow the Lagrangian model, undergo impact, and then once again evolve according to the Lagrangian model. To formulate the trajectory designs and the closed-loop system, we need to split the continuous phase of the model (6.59) into part-(i), after mid-stance, and part-(ii), the first half of the ensuing stance phase.

$$\Sigma_{\rm i}: \begin{cases} \dot{\tau} = 1, & \\ \dot{x} = f(x,u), & x^- \notin \mathcal{S} \\ \tau^+ = \tau^- & \\ x^+ = \Delta(x^-), & x^- \in \mathcal{S} \end{cases}$$

$$\Sigma_{\rm ii}: \begin{cases} \dot{\tau} = 1, & \tau^- < T_p \\ \dot{x} = f(x,u), & \\ \tau^+ = 0 & \tau^- = T_p \\ x^+ = x^-. & \end{cases} \tag{6.61}$$

The guard condition on the phase-i depends only on the state x, whereas the guard condition on the phase-ii depends only on "time" as measured by τ.

H-3 The user has selected an open ball $B \subset \mathbb{R}^n$ about $x^*_{\rm m}$, a positive-definite, locally Lipschitz-continuous function $V : B \to \mathbb{R}$, and constants $0 < \alpha_1 \leq \alpha_2$ such that, $V(x^*_{\mathrm{m}}) = 0$ and $\forall\ x \in B$,

$$\begin{aligned} &\alpha_1 (x - x^*_{\mathrm{m}})^\top (x - x^*_{\mathrm{m}}) \leq \\ &V(x) \leq \alpha_2 (x - x^*_{\mathrm{m}})^\top (x - x^*_{\mathrm{m}}). \end{aligned}$$

H-4 There is a constant $0 \leq c < 1$, such that, for each initial condition $\xi \in B$, there exists a piecewise continuous input $u_\xi : [0, T_p] \to \mathbb{R}^m$ and a corresponding solution $\varphi_\xi : [0, T_p] \to \mathbb{R}^n$ of the hybrid model (6.61) satisfying

a) $\varphi^+_\xi(0) = \xi$,

b) $\varphi^-_\xi({}^{T_p}\!/_2) \in \mathcal{S}$,

c) $\forall\ t \neq {}^{T_p}\!/_2, \varphi_\xi(t) \notin \mathcal{S}$,

d) $\varphi^+_\xi(T_P) \in B$ and there is exponential convergence toward the periodic orbit, namely,

$$V(\varphi^+_\xi(T_P)) \leq cV(\xi), \tag{6.62}$$

and

e) $\xi = x^*_{\mathrm{m}} \Rightarrow u_\xi = u^*$.

■

Proposition VI.11. *Assume the open-loop hybrid model (6.59) satisfies Assumptions H-1 to H-4. Assume in addition there exist open sets B^e_{i} and B^e_{ii} that contain B, a $\delta > 0$, and two feedbacks*

$$\begin{aligned} \mu_{\mathrm{i}} &: [0, {}^{T_p}\!/_2 + \delta] \times B^e_{\mathrm{i}} \to \mathbb{R}^m \\ \mu_{\mathrm{ii}} &: [{}^{T_p}\!/_2 - \delta, T_p] \times B^e_{\mathrm{ii}} \to \mathbb{R}^m \end{aligned}$$

that are piecewise continuous in t, locally Lipschitz continuous in x, and, such that, *for* $0 \leq t < T_p$ *and* $\xi \in B$*,*

$$\begin{aligned} \mu_{\mathrm{i}}(t, \varphi_\xi(t)) &= u_\xi(t), \quad 0 \leq t < {}^{T_p}\!/_2 \\ \mu_{\mathrm{ii}}(t, \varphi_\xi(t)) &= u_\xi(t), \quad {}^{T_p}\!/_2 \leq t < T_p. \end{aligned} \tag{6.63}$$

Then $\varphi^* : [0, T_p] \to \mathbb{R}^n$ *is a locally exponentially stable periodic solution of the closed-loop system*

$$\begin{aligned} &\Sigma_{\mathrm{i}}^{cu} : \begin{cases} \dot{\tau} = 1, & \\ \dot{x} = f(x, \mu_{\mathrm{i}}(\tau, x)), & x^- \notin \mathcal{S} \\ \tau^+ = \tau^- & \\ x^+ = \Delta(x^-), & x^- \in \mathcal{S} \end{cases} \\ &\Sigma_{\mathrm{ii}}^{cu} : \begin{cases} \dot{\tau} = 1, & \tau^- < T_p \\ \dot{x} = f(x, \mu_{\mathrm{ii}}(\tau, x)), & \\ \tau^+ = 0 & \tau^- = T_p \\ x^+ = x^-. & \end{cases} \end{aligned} \tag{6.64}$$

■

6.3.3 Generalized Hybrid Zero Dynamics

Following Appendix-B, assume now that the continuous phase of the hybrid model has been decomposed as

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, u) \\ \dot{x}_2 &= f_2(x_1, x_2, u), \end{aligned} \tag{6.65}$$

with

$$x_2 = \begin{bmatrix} x_{2a} \\ x_{2b} \end{bmatrix} \text{ and } f_2 = \begin{bmatrix} x_{2b} \\ u \end{bmatrix}.$$

Let $\gamma : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$ be a locally Lipschitz continuous insertion function that preserves the periodic orbit, namely, writing $x^*_m =: (x^*_{1m}; x^*_{2m})$, it follows that $\gamma(x^*_{1m}) = x^*_{2m}$.

H-5 There is a constant $0 \leq c < 1$, such that, for each initial condition $\xi = (\xi_1, \gamma(\xi_1)) \in B$, there exists a continuous input $u_{\xi_1} : [0, T_p] \to \mathbb{R}^m$ and a corresponding solution $\varphi_{\xi_1} : [0, T_p] \to \mathbb{R}^n$ of the hybrid model (6.61) satisfying,

a) $\varphi^+_{\xi_1}(0) = (\xi_1; \gamma(\xi_1))$,

b) $\varphi^-_{\xi_1}({}^{T_p}/_2) \in \mathcal{S}$,

c) $\forall\, t \neq {}^{T_p}/_2, \varphi_{\xi_1}(t) \notin \mathcal{S}$,

d) $\varphi^+_{\xi}(T_P) \in B$ and there is exponential convergence toward the periodic orbit, namely,

$$V((\varphi^+_{1\xi_1}(T_p), \gamma(\varphi^+_{1\xi_1}(T_p))) \leq cV(\xi_1, \gamma(\xi_1)), \tag{6.66}$$

and

e) $(\xi_1, \gamma(\xi_1)) = x^*_{\mathrm{m}} \Rightarrow u_{\xi_1} = u^*$.

where a solution of the (n_1+n_2)-dimensional model (6.65) has been decomposed as $\varphi_{\xi_1}(t) =: (\varphi_{1\xi_1}(t), \varphi_{2\xi_1}(t))$.

■

The following result generalizes the hybrid zero dynamics defined in [122, 80, 125]. Even in the case of one degree of underactuation, one is able to achieve exponential stability with this method for gaits that could not be rendered stable with the previous formulation of virtual constraints. See Appendix D for an example. More important that this fact, however, the new formulation allows a systematic approach to robot models with more than one degree of underactuation. This is illustrated in Sect. 6.4.

Proposition VI.12. *Assume the open-loop hybrid system* (6.59) *with* f *given by* (6.65) *satisfies Hypotheses H-1 to H-3 and H-5, and define* $B_1 := \{\xi_1 \in \mathbb{R}^{n_1} \mid (\xi_1, \gamma(\xi_1)) \in B\}$. *Assume in addition there exist open sets* $B^e_{1.\mathrm{i}}$ *and* $B^e_{1.\mathrm{ii}}$ *that contain* B_1, *a* $\delta > 0$, *and two feedbacks*

$$\begin{aligned}\nu_{\mathrm{i}} &: [0, {}^{T_p}\!/_2 + \delta] \times B^e_{1.\mathrm{i}} \to \mathbb{R}^{n_1}\\ \nu_{\mathrm{ii}} &: [{}^{T_p}\!/_2 - \delta, T_p] \times B^e_{1.\mathrm{ii}} \to \mathbb{R}^{n_1}\end{aligned}$$

and

$$\begin{aligned}\mu_{\mathrm{i}} &: [0, {}^{T_p}\!/_2 + \delta] \times B^e_{1.\mathrm{i}} \to \mathbb{R}^{m}\\ \mu_{\mathrm{ii}} &: [{}^{T_p}\!/_2 - \delta, T_p] \times B^e_{1.\mathrm{ii}} \to \mathbb{R}^{m}\end{aligned}$$

that are piecewise continuous in t, *locally Lipschitz continuous in* x, *and, such that, for* $0 \le t < T_p$ *and* $\xi \in B$,

$$\begin{aligned}\nu_{\mathrm{i}}(t, \varphi_{1\xi_1}(t)) &= \varphi_{2\xi_1}(t), \quad 0 \le t < {}^{T_p}\!/_2\\ \nu_{\mathrm{ii}}(t, \varphi_{1\xi_1}(t)) &= \varphi_{2\xi_1}(t), \quad {}^{T_p}\!/_2 \le t < T_p\end{aligned} \tag{6.67}$$

and

$$\begin{aligned}\mu_{\mathrm{i}}(t, \varphi_{1\xi_1}(t)) &= u_{\xi_1}(t), \quad 0 \le t < {}^{T_p}\!/_2\\ \mu_{\mathrm{ii}}(t, \varphi_{1\xi_1}(t)) &= u_{\xi_1}(t), \quad {}^{T_p}\!/_2 \le t < T_p.\end{aligned} \tag{6.68}$$

Then $x_1^* : [0, T_p] \to \mathbb{R}^{n_1}$ *is a locally exponentially stable periodic solution of the* *reduced-order hybrid system*

$$\Sigma_{\mathrm{i}} : \begin{cases} \dot{\tau} = 1, \\ \dot{x}_1 = f_1(x_1, \nu_{\mathrm{i}}(\tau, x_1), \mu_{\mathrm{i}}(\tau, x_1)), \\ \quad \textit{when} \begin{bmatrix} x_1^- \\ \nu_{\mathrm{i}}(\tau^-, x_1^-) \end{bmatrix} \notin \mathcal{S} \\ \tau^+ = \tau^-, \\ x_1^+ = \Delta_1(x_1^-, \nu_{\mathrm{i}}(\tau^-, x_1^-)), \\ \quad \textit{when} \begin{bmatrix} x_1^- \\ \nu_{\mathrm{i}}(\tau^-, x_1^-) \end{bmatrix} \in \mathcal{S} \end{cases} \tag{6.69}$$

$$\Sigma_{\mathrm{ii}} : \begin{cases} \dot{\tau} = 1, \qquad \tau^- < T_p \\ \dot{x}_1 = f(x, \nu_{\mathrm{ii}}(\tau, x_1), \mu_{\mathrm{ii}}(\tau, x_1)), \\ \tau^+ = 0, \qquad \tau^- = T_p \\ x^+ = x^-. \end{cases}$$

■

Remark VI.13.

(i) In principle, $\tau^* : [0, T_p] \to \mathbb{R}$ needs to be defined to complete the periodic orbit, but clearly, the trivial solution, $\tau^*(t) = t$, is the only possibility.

(ii) As in the non-hybrid case, using the trajectories in H-5, define

$$\Psi_t : B_1 \to \mathbb{R}^n, \ \Psi_t(\xi_1) := \begin{bmatrix} \varphi_{1\xi_1}(t) \\ \varphi_{2\xi_1}(t) \end{bmatrix} \tag{6.70}$$

and $\Psi_e : [0, T_p) \times B_1 \to \mathbb{R}^{n+1}$ by

$$\Psi_e(\tau, \xi_1) := \begin{bmatrix} \tau \\ \Psi_\tau(\xi_1) \end{bmatrix}. \tag{6.71}$$

By H-5-b), $\forall\ \xi_1 \in B_1$, $\Psi_e^-({}^{T_p}\!/_2, \xi_1) \in \mathcal{S}$. Hence, the loss of dimension is in the τ-component, and therefore

$$\dim \left(\Psi_e^-({}^{T_p}\!/_2, B_1) \cap \mathcal{S}\right) = \dim(B_1).$$

(iii) The Generalized Hybrid Zero Dynamics Manifold (G-HZD) is therefore[3]

$$Z_e := \Psi_e([0, T_p), B_1), \tag{6.72}$$

which has two components,

$$Z_{e,i} := \Psi_e([0, {}^{T_p}\!/_2), B_1)$$

and

$$Z_{e,ii} := \Psi_e([{}^{T_p}\!/_2, T_p), \Psi_e^-({}^{T_p}\!/_2, B_1) \cap \mathcal{S}).$$

(iv) The corresponding restriction dynamics is given by (6.69), which is then the G-HZD.

[3]Modulo T_p is not required here because τ is reset at T_p, whereas in the non-hybrid case, it was required in (6.28).

6.3.4 Stabilizing the Original Model

We can now obtain and explain the controller we use on bipeds. Similar to Sect. 6.1.5, assume the continuous phase of the hybrid model has the form

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, u_1) \\ \dot{x}_2 &= f_2(x_1, x_2, u_2), \end{aligned} \tag{6.73}$$

with

$$x_2 = \begin{bmatrix} x_{2a} \\ x_{2b} \end{bmatrix} \text{ and } f_2 = \begin{bmatrix} x_{2b} \\ u_2 \end{bmatrix},$$

and $u = (u_1, u_2)$. The reason to split the input and not allow the u_2-component to enter the x_1-dynamics will be clear shortly. We allow the u_1-component to be empty.

H-6 The solutions in H-5 also satisfy

$$\gamma(\varphi_{1\xi_1}(T_p)) = \varphi_{2\xi_1}(T_p). \tag{6.74}$$

■

Theorem VI.14. *Assume the open-loop hybrid system (6.59) with f given by (6.73) satisfies Hypotheses H-1 to H-3, H-5 and H-6. Define $B_1 := \{\xi_1 \in \mathbb{R}^{n_1} \mid (\xi_1, \gamma(\xi_1)) \in B\}$. Assume in addition there exist open sets $B^e_{1.\mathrm{i}}$ and $B^e_{1.\mathrm{ii}}$ that contain B_1, a $\delta > 0$, and feedbacks*

$$\begin{aligned} \nu_{\mathrm{i}} &: [0, {}^{T_p}\!/_2 + \delta] \times B^e_{1.\mathrm{i}} \to \mathbb{R}^{n_1} \\ \nu_{\mathrm{ii}} &: [{}^{T_p}\!/_2 - \delta, T_p] \times B^e_{1.\mathrm{ii}} \to \mathbb{R}^{n_1} \end{aligned}$$

and

$$\mu_{\mathrm{i}} : [0, {}^{T_p}\!/_2 + \delta] \times B^e_{1.\mathrm{i}} \to \mathbb{R}^m$$
$$\mu_{\mathrm{ii}} : [{}^{T_p}\!/_2 - \delta, T_p] \times B^e_{1.\mathrm{ii}} \to \mathbb{R}^m$$

that are piecewise continuous in t*, locally Lipschitz continuous in* x*, and, such that, for* $0 \le t < T_p$ *and* $\xi_1 \in B_1$*,*

$$\begin{aligned} \nu_{\mathrm{i}}(t, \varphi_{1\xi_1}(t)) &= \varphi_{2\xi_1}(t), \quad 0 \le t < {}^{T_p}\!/_2 \\ \nu_{\mathrm{ii}}(t, \varphi_{1\xi_1}(t)) &= \varphi_{2\xi_1}(t), \quad {}^{T_p}\!/_2 \le t < T_p \end{aligned} \tag{6.75}$$

and

$$\begin{aligned} \mu_{\mathrm{i}}(t, \varphi_{1\xi_1}(t)) &= u_{\xi_1}(t), \quad 0 \le t < {}^{T_p}\!/_2 \\ \mu_{\mathrm{ii}}(t, \varphi_{1\xi_1}(t)) &= u_{\xi_1}(t), \quad {}^{T_p}\!/_2 \le t < T_p. \end{aligned} \tag{6.76}$$

Then for all $\frac{n_2}{2} \times \frac{n_2}{2}$ *positive definite matrices* K_p *and* K_d*,* $\exists\ \epsilon^* > 0$*, such that* $\forall\ 0 < \epsilon \le \epsilon^*$*,* $x^* : [0, T_p] \to \mathbb{R}^{n_1+n_2}$ *is a locally exponentially stable periodic solution* *of the closed-loop hybrid system*

$$
\Sigma_{\mathrm{i}}:\begin{cases}
\dot{\tau} = 1 & \\
\dot{x} = f(x_1, x_2, u_1, u_2) & x^- \notin \mathcal{S} \\
u_1 = \mu_{1i}(\tau, x_1) & \\
u_2 = \mu_{2i}(\tau, x_1) - \left[\dfrac{K_p}{\epsilon^2}\,\dfrac{K_d}{\epsilon}\right]\big(x_2 - \nu_i(\tau, x_1)\big) & \\
\tau^+ = \tau^- & \\
x^+ = \Delta(x^-), & x^- \in \mathcal{S}
\end{cases}
\tag{6.77}
$$

$$
\Sigma_{\mathrm{ii}}:\begin{cases}
\dot{\tau} = 1 & \tau^- < T_p \\
\dot{x} = f(x_1, x_2, u_1, u_2) & \\
u_1 = \mu_{1ii}(\tau, x_1) & \\
u_2 = \mu_{2ii}(\tau, x_1) - \left[\dfrac{K_p}{\epsilon^2}\,\dfrac{K_d}{\epsilon}\right]\big(x_2 - \nu_{ii}(\tau, x_1)\big) & \\
\tau^+ = 0 & \tau^- = T_p \\
x^+ = x^-. &
\end{cases}
$$

Moreover, the closed-loop system possesses a G-HZD and it is given by (6.69).

Remark VI.15. The control was split so that the high-gain part of the feedback does not directly enter the states of the zero dynamics, namely x_1. This allows the system to conform with existing theorems [4] for establishing the exponential stability of the periodic orbit — in the full-order hybrid model — on the basis of its stability in the zero dynamics, (6.69). Isolating the action of the high-gain controller to the x_2-dynamics was not necessary in the case of ODEs.

6.4 Bipedal Walking Gaits

This section applies the Generalized Hybrid Zero Dynamics (G-HZD) developed in Section 6.3 to the bipedal robot MARLO. The control laws developed here will illustrate stepping in place, walking forward and backward, and transitions among such gaits. The work illustrates a theoretically sound method for gait design that unifies and significantly extends many of our previous results. The first control designs will rely on the optimization package in [58], which can only handle a planar model of the robot. For these designs, lateral stability is achieved via a heuristic foot placement policy. Since March of 2017, we have had access to the trajectory optimization package [53], which easily handles the full 3D model of the robot. Figure 6.16 summarizes our control design process.

6.4.1 MARLO Model Update

The robot is described in detail in [97]. For the planar model, the configuration variables are joint angles and one absolute coordinate. The angle θ, the absolute stance leg angle, is unactuated because the feet are passive. Hence, we define

$$x_1 = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}.$$

For the purposes of controller design, the regulated quantities are

$$q_a = (q_x, q_{sw,LA}, q_{st,KA}, q_{sw,KA}),$$

that is, the torso, stance knee, swing leg, and swing knee angle, and hence

$$x_2 = \begin{bmatrix} q_a \\ \dot{q}_a \end{bmatrix}.$$

In the simulation and control design, we constrain the stance foot to remain in contact with the ground with no foot slip. In the experiments, we estimate the ground reaction forces through the deflection of the leaf springs to decide whether to control the torso angle or the stance leg angle $q_{st,LA}$, as in [104]. The model decomposition is done as in Appendix B.

6.4.2 Design of Planar Periodic Gaits and Transition Trajectories via Optimization

For robots, an orbit library is called a gait library. We first design a gait library

$$\mathcal{L} := \{\bar{v} \mid -0.8 \leq \bar{v} \leq 0.8\} \tag{6.78}$$

consisting of periodic gaits for various average walking speeds satisfying H-2. In this example, we reuse the gaits described in [33], where each gait has period $T_p = 0.4$, and the cost function is

$$J = \int_{-T_p/2}^{T_p/2} ||u(t)||^2 dt. \tag{6.79}$$

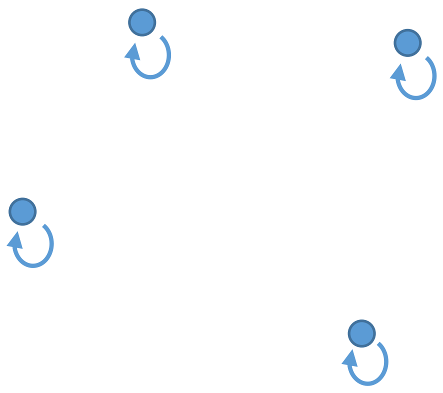
Denote the trajectory of the periodic gait and the corresponding input as $\varphi^{\bar{v}}(t)$ and $u^{\bar{v}}(t)$, respectively, and the midpoint of the periodic trajectory as $x_m^{\bar{v}}$. The insertion function is built from the gait library and is denoted $\gamma_{\mathcal{L}}$.

For a given periodic orbit in $\mathcal{L}$, we define

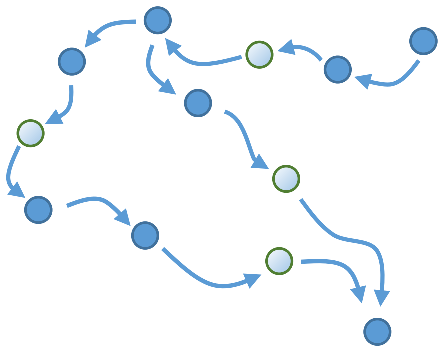
$$\begin{aligned} B_1^{\bar{v}} = \{\xi_1 := (\theta, \dot{\theta}) \mid -\frac{\pi}{12} \leq |\theta - \theta_m^{\bar{v}}| \leq \frac{\pi}{12}, \\ -0.8 \leq |\dot{\theta} - \dot{\theta}_m^{\bar{v}}| \leq 0.8\}, \end{aligned} \tag{6.80}$$

a sliding window[4] about the target speed $\bar{v}$. For $\xi_1 \in B_1^{\bar{v}}$, trajectories are generated

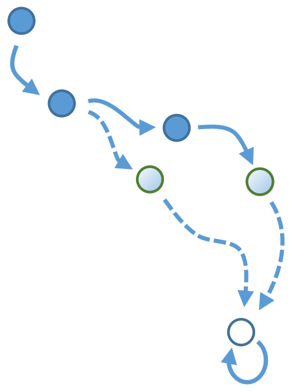
[4]Because the legs are 1 m long, average walking speed and angular rate at the middle of the step are nearly the same.



Step 1: Generate **periodic gaits**



Step 2: Design **transient gaits** between periodic gaits; here, we are showing three-step transitions. Similar to MPC, only the first step of the transition is retained for the learning set. From this set of trajectories, **Supervised Machine Learning** is used to extract the controller



Step 3: The hollow dot is the target periodic gait. It can be reached in three steps, though the **learned controller** re-plans at each step to reach the target in three additional steps, leading to exponential convergence.

Figure 6.16: This can be thought of as an alternative representation of Figure 1.3 when the surface Z_0 is built from periodic solutions of the full-order model. The light dots represent transient trajectories while the other dots (solid or hollow) are periodic orbits.

Table 6.4: Optimization Constraints

Motor Toque $\lvert u \rvert$	< 5 Nm
Step Duration T	$= 0.4$ s
Friction Cone μ	< 0.6
Impact Impulse Fe	< 15 Ns
Vertical Ground Reaction Force	> 300 N
Mid-step Swing Foot Clearance	> 0.15 m

as in H-5 using optimization with cost function

$$J(\xi_1, \bar{v}) = \sum_{k=1}^{6} \int_{(k-1)\frac{T_p}{2}}^{k\frac{T_p}{2}} \left(||x - \varphi^{\bar{v}}||_Q^2 + ||u - u^{\bar{v}}||_R^2\right) dt \tag{6.81}$$

for a horizon of length $T_h = 3T_p$. The optimization is performed subject to the hybrid dynamics describing MARLO, the physical constraints shown in Table. 6.4, and the terminal constraint $x(3T_p) = \varphi^{\bar{v}}(T_p)$. So that the trajectories can be used to stabilize the full-order model, the boundary constraints in H-6 is also imposed. The resulting trajectory and input for the given $\bar{v}$ and selected ξ_1 are denoted $\varphi_{\xi_1}^{\bar{v}}(t)$ and $u_{\xi_1}^{\bar{v}}(t)$, respectively.

Remark VI.16. The initial condition set $B_1^{\bar{v}}$ is related to the notion of 3-step capture region defined in [67, 131]. In our experience, three-steps is a reasonable balance between planning horizon and computational burden.

6.4.3 Controller Design via Machine Learning

The gait library (6.78) is assumed to be discretized by 5 evenly spaced average speeds $\bar{v}_k$, each $\xi_1^i =: (\theta^i, \dot{\theta}^i)$ is drawn from a uniformly spaced gird of 25 points, and time interval $[0, T_p]$ is evenly sampled into 21 points, t_j. The combined training and validation data set is therefore denoted by

$$\begin{aligned} x_1^{i,j,k} &:= \varphi^{\bar{v}_k}_{1\xi_1^i}(t_j) \\ \nu^{i,j,k} &:= \varphi^{\bar{v}_k}_{2\xi_1^i}(t_j) \\ \mu^{i,j,k} &:= u^{\bar{v}_k}_{\xi^i}(t_j). \end{aligned} \tag{6.82}$$

Any infeasible optimization problems[5] are removed from the data set before processing it by Supervised Machine Learning.

We next learn the functions in Theorem VI.14. The features are $(t_j, x_1^{i,j,k}, \bar{v}_k)$ and the labels are $(\nu^{i,j,k}, \mu^{i,j,k})$ and the data base is split at $t = {}^{T_p}/_2$ so that the functions

$$\nu_{\mathrm{i}}(t, x_1, \bar{v})$$
$$\nu_{\mathrm{ii}}(t, x_1, \bar{v})$$
$$\mu_{\mathrm{i}}(t, x_1, \bar{v})$$
$$\mu_{\mathrm{ii}}(t, x_1, \bar{v})$$

are learned individually. Part of the fitting is shown in Figure 6.17. These functions are enough to construct the G-HZD in (6.69). To complete the control design as in (6.77), the feedback gains K_p, K_d and ϵ must be selected. While in principle these last gains may have to vary with $\bar{v}$, for MARLO, we have found that a single set of gains[6] suffices. The transition among different target speeds are shown in Figure 6.18.

6.4.4 Example Performance Analysis

6.4.4.1 Stability Analysis

We know that the periodic orbits in the full-order model should be locally exponentially stable by the results in Sect. 6.3. We formally verify this by numerically

[5] For example, due to torque limits, there is no solution for $\bar{v} = 0.8$ and $\xi_1 = (\theta_m^{\bar{v}} + {}^{\pi}/_{12}, 0.8 + \dot{\theta}_m^{\bar{v}})$.

[6] They essentially correspond the low-level PD gains which are straightforward to tune on MARLO. In simulation, $K_p = 800$, $K_d = 40$ and $\epsilon = 1$

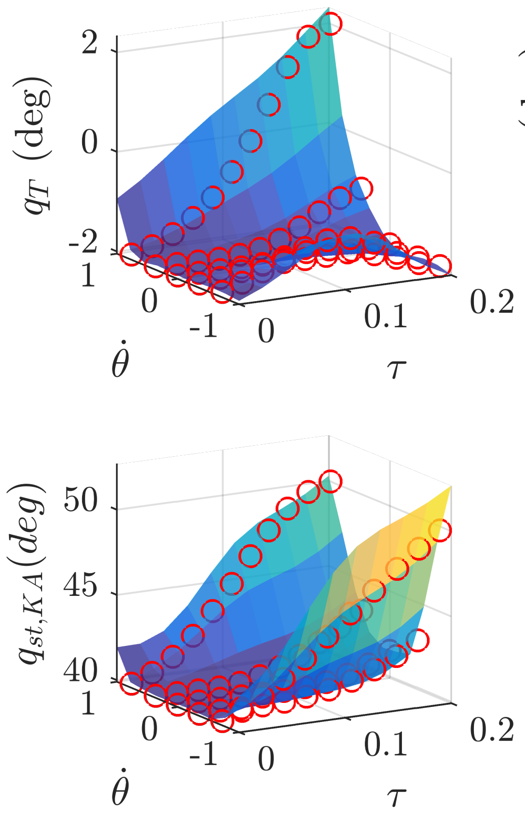


Figure 6.17: The fitting results of the Supervised Machine Learning. The features $\theta = \theta_m^{\bar{v}}$ and $\bar{v} = 0$ are fixed at constant values, while $\tau \in [0, 0.2]$ is from mid-step to ground contact. The plots show four of the components in $\nu_{\mathrm{i}}(t, x_1, \bar{v})$.

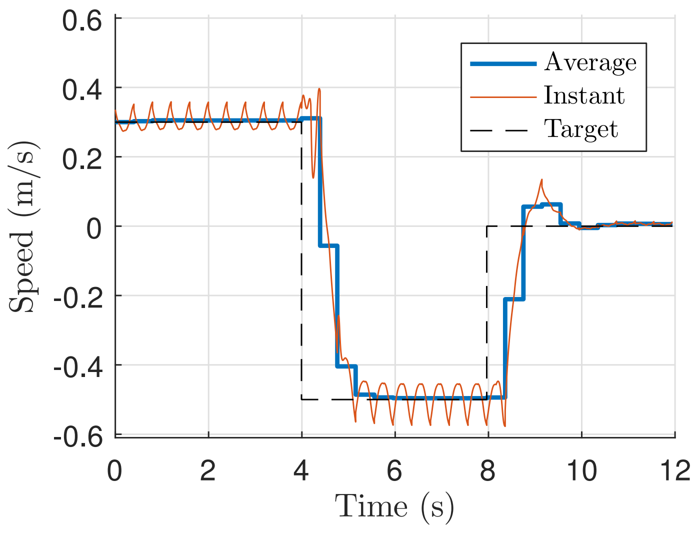


Figure 6.18: The target speed $\bar{v}$ changes from 0.3 m/s to -0.5 m/s and to 0 m/s. The gait transition takes less than five steps to reach the target speed. The error between target and average speed is small.

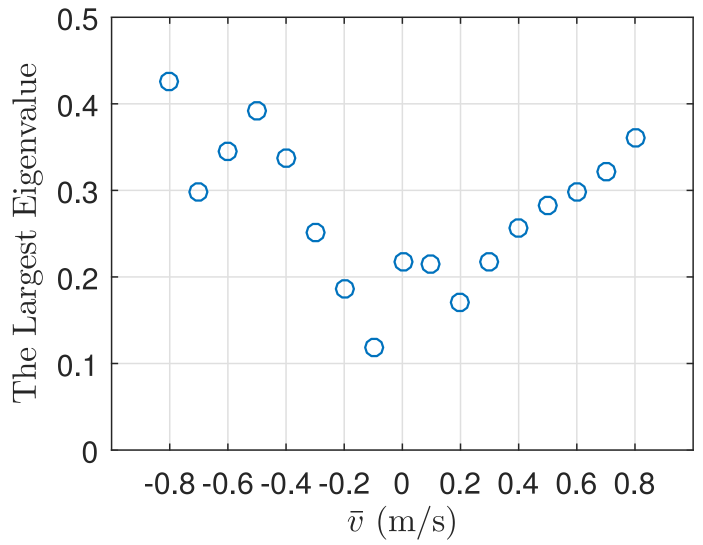


Figure 6.19: The largest eigenvalue of the Poincaré map is given for some target speeds. This indicates the target speed in the gait library (6.78) is exponentially stable.

evaluating the Jacobian of the Poincaré map for twenty evenly-spaced points in the interval $-0.8 \leq \bar{v} \leq 0.8$. The magnitude of the largest eigenvalue is shown in Figure 6.19, which proves local exponential stability for each fixed target speed $\bar{v}$. Note that only five of these points were in the training data. The learned feedback functions have provided stable gaits for a continuum of target speeds.

The stability of the overall closed-loop system is further illustrated by applying force perturbations, which is a more “realistic” test. We apply a longitudinal force on the hip at 1.2 seconds for 0.8 second (i.e., $2T_p$) and examine the time to recover the nominal gait. For the stepping-in-place gait $\bar{v} = 0$, the largest force from which the robot can recover without violating the physical constraints is 150 N. Figure 6.20 shows the resulting longitudinal velocity of the robot. The peak speed is approximately 1.5 m/s, which is beyond the maximum training speed of 0.8 m/s. The speed is once again less than 0.05 m/s within five steps. The convergence rate is relatively fast given that the optimizer uses a horizon of three steps.

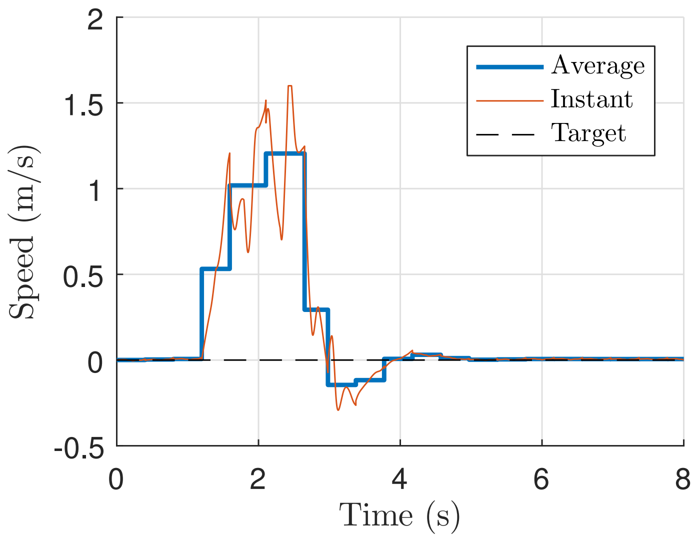


Figure 6.20: A perturbation is applied from 1.2 seconds to 2 seconds in a magnitude of 150 N. The maximum speed is larger than the maximum speed (0.8 m/s) in the training set. The extrapolation may be the cause of the speed oscillation. Even though, the speed convergence near the target speed in less than five steps.

6.4.4.2 Interpretation of the posture changes employed by the controller

We now provide some physical intuition for how the controller coordinates the links to achieve stability. In fact, we evaluate $\nu({}^{T_p}\!/_2, x_1)$ with $x_1 = ({}^{\pi}\!/_{12}, \dot{\theta})$, $-0.8 \leq \dot{\theta} \leq 0.8$. Figures 6.21 and 6.22 show the changes in the swing leg angle and the stance knee angle at touchdown. The swing leg is seen to obey an approximately linear relationship with respect to velocity, just as in the foot placement controllers in [94, 36] designed on the basis of an inverted pendulum model or a linear inverted pendulum model, viz

$$\Delta q_{sw,LA} = K_1 \Delta v,$$

and the scalar K_1 is constant. Denote the regressed linear fit in Figure 6.21 by K_1^*. We add $\pm 10\%$ to K_1^* and compare the resulting foot placement strategies in Figure 6.23. It is seen that with the smaller gain the velocity takes longer to settle whereas with the larger gain, there is overshoot.

The learned controller is more than just providing foot placement. Figure 6.22 also

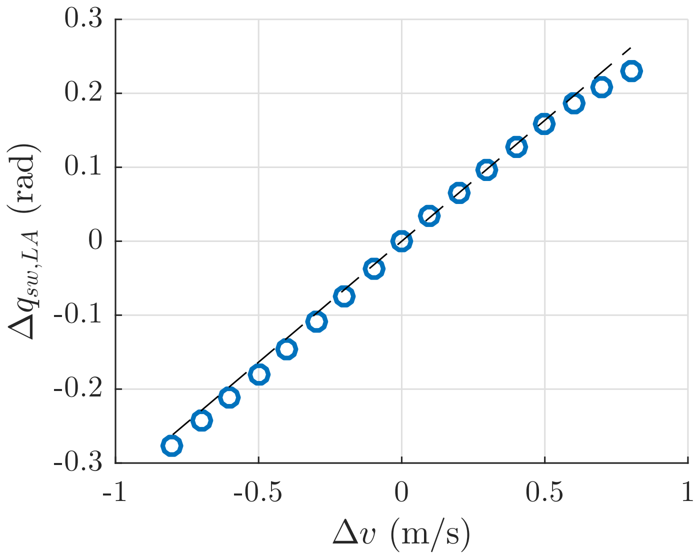


Figure 6.21: Change in swing leg angle vs change in velocity. One part of the learned-optimal strategy is a standard linear leg-angle adjustment policy as in [94, 36].

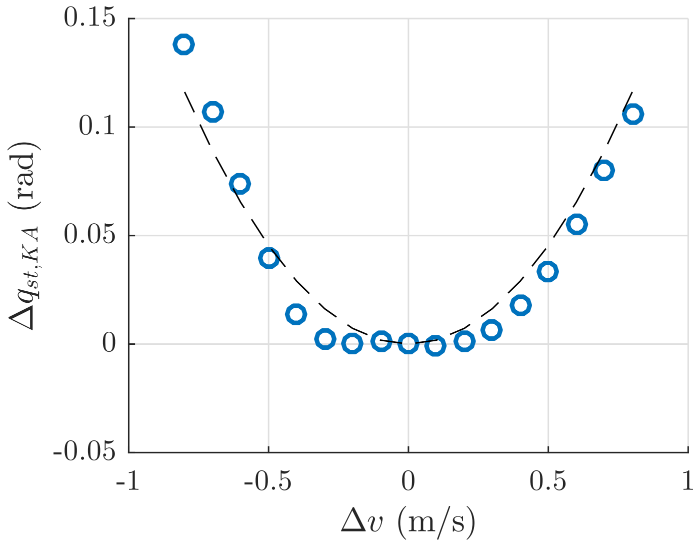


Figure 6.22: Change in stance knee angle vs change in velocity. This is not part of the standard recommendations in [94, 36].

shows a quadratic relationship for knee angle versus velocity just before touchdown, viz

$$\Delta q_{st,KA} = K_2(\Delta v)^2.$$

As the velocity moves away from zero in either direction, the stance knee angle increases. Perhaps this is to lower the center of mass and to make it easier for the swing

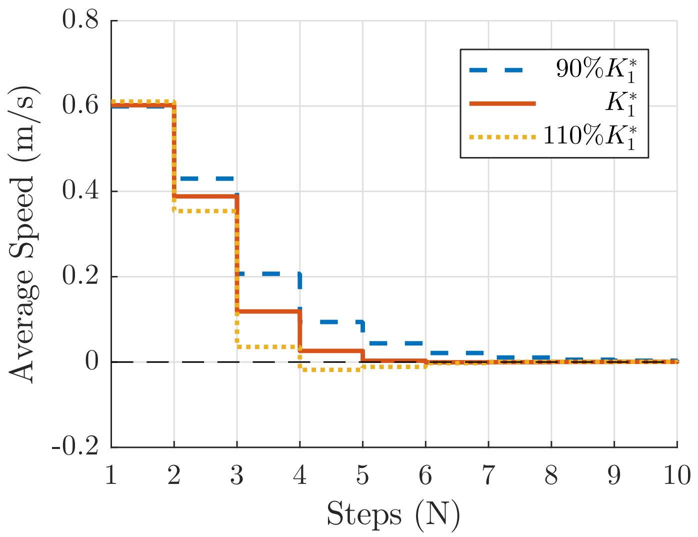


Figure 6.23: When the effective linear policy from the learned-optimal strategy is modified by ±10%, the convergence to the nominal speed of zero either slows down or overshoots.

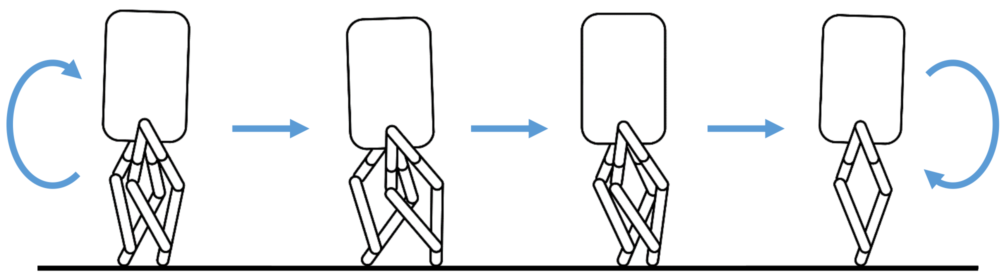


Figure 6.24: Stick figure showing the coordinated action of torso angle, knee bend, and leg angle provided by the learned-three-step optimization.

foot to touch the ground. Furthermore, in addition to changing the swing leg angle, the learned controller also straightens the swing knee angle, thereby extending the foot further out. Finally, it also leans the torso backward, keeping the center of mass over the stance toe, as shown in Figure 6.24. These adjustments are all coordinated by the optimization and automatically extracted from the transition trajectories by the supervised learning. Unlike a classical foot placement controller that adjusts only swing leg angle, the learned controller uses the many degrees of freedom of the robot to achieve better performance.

6.5 Experiments on a 3D Bipedal Robot

This section extends the learning controller of the last section to the full-3D model of MARLO. Hence, both the sagittal and lateral planes of the robot are included, while yaw rotations are assumed to be small due to the foot. This control design will allow the physical 3D-robot to walk and step in place. The 3D-controller design will mostly follow the process of the planar example that was illustrated through simulations, though some modifications have been made during the experimental implementation to deal with model uncertainty, impact model uncertainty, and signal noise; these will be clearly explained and justified.

6.5.1 Update 3D-MARLO Configuration

We update the generalized coordinates in Figure 3.2 to describe the 3D robot. Let (p_x, p_y) denote the sagittal and lateral position of the center of mass, and be (v_x, v_y) be the corresponding velocities. We define

$$x_1 = \begin{bmatrix} p_x \\ p_y \\ v_x \\ v_y \end{bmatrix}$$

as the "weakly actuated" states. The regulated angles are $q_a = (q_y, q_x, q_{sw,3}, q_{sw,LA}, q_{st,KA}, q_{sw,KA})$, that is, the torso roll and pitch, swing hip, swing leg, stance knee and swing knee angles, and hence the "strongly actuated" states are

$$x_2 = \begin{bmatrix} q_a \\ \dot{q}_a \end{bmatrix}.$$

We note that the coordinates (x_1, x_2) describe the robot dynamics in (6.65), or (B.6) after a coordinate change.

6.5.2 Optimization

We first use optimization, with a cost function as in (6.79), to build a periodic gait library

$$\begin{aligned} \mathcal{L} := \{(\bar{v}_x, \bar{v}_y) \mid & -0.6 \leq \bar{v}_x \leq 0.6, \\ & -0.4 \leq \bar{v}_y \leq 0.4\}, \end{aligned} \tag{6.83}$$

as a grid in two-dimensional Cartesian space; i.e., longitudinal and lateral speed of the robot. The gaits are designed, without loss of generality, such that the associated 2D-Cartesian positions (p_x, p_y) are equal to the origin at a nominal point in each of the gaits. The insertion function associated to the gait library is constructed using linear regression as in the pendulum model (6.55) and in the planar biped examples; specifically,

$$x_2 = \gamma_{\mathcal{L}}(x_1) = a_0 + a_1 x_1, \tag{6.84}$$

where a_0 and a_1 are constant vectors. A linear fitting is good enough for 3D MARLO. While one could do a more sophisticated fit, the maximum root-mean-square-error (RMSE) for all joints is less than 1 degree, and for all joint velocities it is less than 4 degrees per second, even with the linear fit. In part, this is a benefit of using the middle of the gait for building the controller.

The next step is to design transition trajectories from periodic orbits in the library $\mathcal{L}$ to a target periodic orbit. In this chapter, we will only illustrate the target orbit as the stepping-in-place gait, that is, $(\bar{v}_x^*, \bar{v}_y^*) = 0$. Further details are not given because they follow the planar example of the last section. One can also refer to work in [33] for the design of transition gaits for different ground slopes.

We perform three-step trajectory optimizations as in (6.81) and denote the collection of transition trajectories to the target speed of zero as $\varphi_{\mathcal{L}\to 0}$. The orbit library is evenly sampled per $\bar{v}_x = \{-0.6, -0.4, -0.2, \ldots, 0.6\}$ and $\bar{v}_x = \{-0.4, -0.3, -0.2, \ldots, 0.4\}$, so that the total number of transition trajectories ${\varphi^i}_{\mathcal{L}\to 0}$ is 63. The time interval $[0, 0.4]$ is evenly sampled into 21 points, t_j.

6.5.3 Machine Learning

For the 3D robot, we illustrate a different philosophy in building the feature set. Recall that in the planar examples, we included time and all of x_1-coordinates in the feature vector. Here, we ill select the feature vector as simply time and the velocity coordinates, namely,

$$(t, v_x, v_y),$$

and leave out the Cartesian positions (p_x, p_y). There are several reasons for this:

1. The impact map in the hybrid model (6.61) resets the Cartesian position variables to a constant, assumed to the origin. Hence, these variables do not need to be stabilized.

2. On the robot MARLO, we are placing the torso sagittal and roll angles in the x_2-coordinates, and hence if these are kept upright, the position of the center of mass does not provide significant additional information.

3. It keeps the dimension of the feature set as low as possible, which allows a smaller training data set.

On 3D MARLO, the labels are taken as

$$\nu = \varphi_{2,\mathcal{L}\to 0},$$

which represents the x_2-coordinates of the optimized trajectories. The control input μ is not learned because, as in previous experiments [48, 33] on this robot, we use PD control

$$u_{PD} = [K_p, K_d]\left(x_2 - \nu(t, v_x, v_y)\right) \tag{6.85}$$

to regulate the joint angles, without a feedforward term. The feedforward torque is not applied because of uncertainty in the model. Specifically, the model does not include the motor drive friction, which consumes about 20% of the torque in nominal operation (stepping in place and walking), nor does the model include backlash or compliance in the harmonic drives. Finally, the leaf springs in Figure 3.2 are excluded from the model; they deflect about 5 degrees when supporting the robot's weight.

Since the impact happens at ${}^{T_p}/_2 = 0.2$, the functions

$$\nu_{\text{i}}(t, v_x, v_y)$$
$$\nu_{\text{ii}}(t, v_x, v_y)$$

are learned individually.

6.5.4 Experimental Implementation

Another difference between the model and the physical robot occurs in the velocity signal. Due to spring deflection, impacts, and joint compliance, the estimated Cartesian velocities (v_x, v_y) are noisy even if each of the individual joint angular velocity signals is relatively "clean". We thus use a strong[7] first-order filter to clean up the Cartesian velocity signals (v_x, v_y) appearing in $\nu(t, v_x, v_y)$, the reference for low-level PD controllers (6.85).

The filtered signal, shown in Figure 6.25, is relatively clean, but causes phase lag. Moreover, the energy loss at impact is less on the robot than in the design model because the springs store energy at impact release it throughout a step. This two factors lead the learned controller to generate overshoot in the Cartesian velocities.

[7]The time constant is 0.2 second.

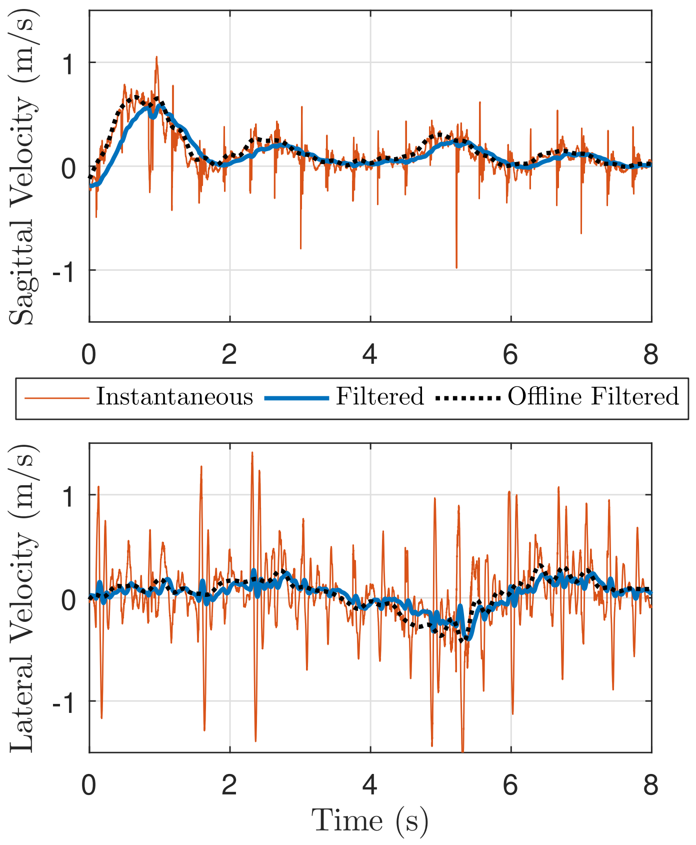


Figure 6.25: A perturbation is applied from 1.2 seconds to 2 seconds in a magnitude of 150 N. The maximum speed is larger than the maximum speed (0.8 m/s) in the training set. The extrapolation may be the cause of the speed oscillation. Even though, the speed convergence near the target speed in less than five steps.

We mitigate the overshoot by introducing a speed-damping term on the torque of the swing leg and the swing hip,

$$\begin{aligned} u_d^{sw,LA} &= N_{x,d}(v_x - v_x^k) \\ u_d^{sw,3} &= N_{x,d}(v_y - v_y^k), \end{aligned} \tag{6.86}$$

which is the same term used in [104]. The overall torque is

$$u = u_{PD} + u_d. \tag{6.87}$$

6.5.5 Results

The learned controller for stabilizing the stepping-in-place gait was implemented on MARLO. The nominal Cartesian velocities are thus zero. Forces were applied in the sagittal and lateral directions, or a mix of the two, by the experimenter applying a push or a kick to the robot. The amount of force has not been estimated, but the reader can judge of his-or-herself based on the experiment video (see also Extension 4).

In the first experiment, five successive kicks were applied to MARLO in the forward (sagittal) direction. MARLO consistently recovered from the disturbances. The peak speed varied from 0.8 to 1 m/s, as shown in Figure 6.26. A harder kick was not applied since the training set only includes speed up to 0.6 m/s. After reaching the peak speed, MARLO slowed down to 0.1 m/s in less than 5 seconds. The robot acted as an underdamped spring-load system. This may be caused by the leg springs in the physical which compressed and unloaded on the second step, while the model did not include this effect. We have added the damping term in (6.86) to mitigate the overshoot effect. A larger derivative gain will further reduce the overshoot, but will so increase the settling time. Still, the speed slowed down to 0.1 m/s in less than 5 seconds. The leg motor torque is shown in Figure 6.26. The torque bound (5 Nm) was reached when robot moved around the peak speed. This could explain why the optimization can only find the solution of transition gait from 0.6 m/s to zero.

The second experiment was to push MARLO in the lateral direction. Because of the parallelogram shape of the legs, one foot cannot place across the other, which limits the available range of foot placement. Plus the weaker motor on the hip, the lateral stability of MARLO is weaker than the sagittal direction. The push drove the lateral speed to 0.6 m/s, which is higher than the training speed 0.4 m/s. The push was applied on both left and right sides, shown in Figure 6.28. The hip motor torque is shown in Figure 6.29. The torque bound is 3 Nm.

Random direction pushes and kicks were included in the last part of the experiment video. We applied force to move MARLO backward and to turn around.

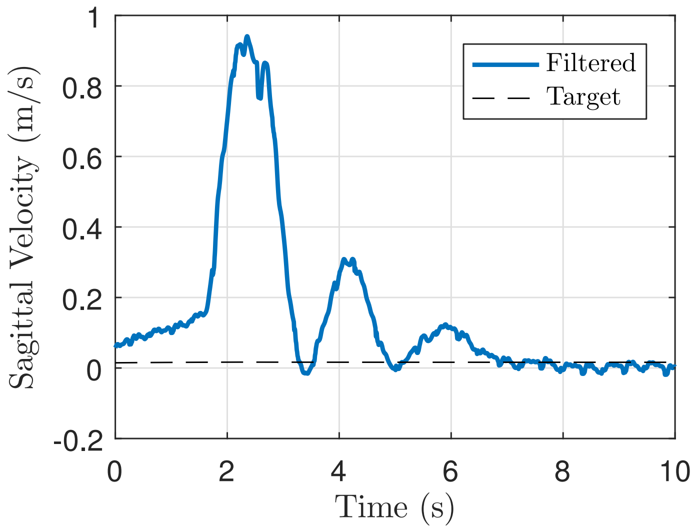


Figure 6.26: An example of the velocity response for a kick in the forward (sagittal) direction. The perturbation is applied at around 1 second driving the robot to peak speed of 0.9 m/s.

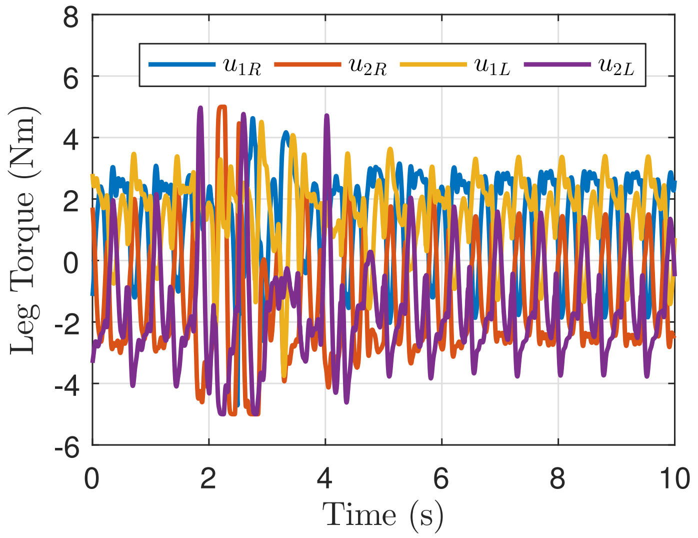


Figure 6.27: The leg torques are applied before a 1:50 gear transmission. The torques $(u_{1R}, u_{2R}, u_{2L}, u_{2L})$ are associate with the leg joints $(q_{1R}, q_{2R}, q_{1L}, q_{2L})$ in the robot configuration, Figure 3.2.

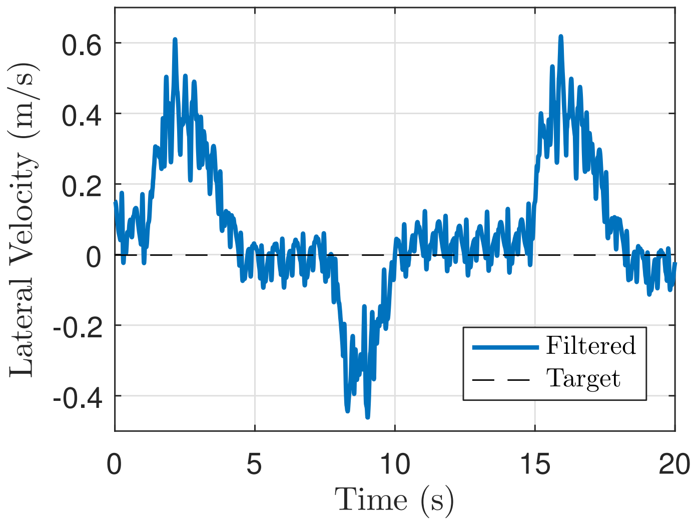


Figure 6.28: An example of the velocity response for multiple pushes in the lateral direction. The positive sign is the right direction whereas the negative is the left.

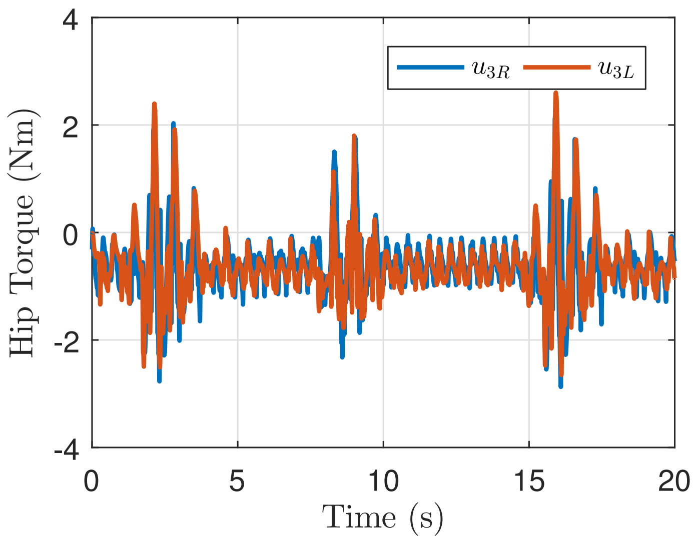


Figure 6.29: The hip torques are applied before a 1:27.5 gear-belt drive. The positive torque is to move leg inward whereas the positive torque is to move leg outward.

CHAPTER VII

Discussion and Conclusion

7.1 Strategy

This thesis is building on the recent revolution in open-loop trajectory optimization. It is now possible to compute in minutes gaits that used to take us hours or more. Armed with a set of open-loop trajectories, the question we posed was, how to turn them into a feedback controller? Our strategy was to attempt to build a surface from the trajectories and to induce a vector field on that surface that had desirable properties, such as (1) it contained a periodic solution of the model that met important physical constraints; (2) trajectories on the surface, by design, converged to the periodic solution; and (3), we could find a feedback controller for the complete model of the system that would render this surface exponentially attractive.

We have used Supervised Machine Learning as a “universal” function approximation, in other words, glorified regression. The functions we sought were implicitly contained in the data, and to our knowledge, closed-form solutions seem unlikely to exist. Hence, they had to be computed numerically in one fashion or another, and we believe an important contribution of the thesis is to show how the functions needed to build a feedback controller can be extracted from a set of trajectories over a fixed time interval.

7.2 Curse of Dimensionality

The method we use to build a vector field from open-loop trajectories works, at least in principle, in large dimensions. Even with the large strides made in optimization, high-dimensional state spaces are still the bottleneck, at least with our approach. Hence, it was important to find a structural property in our robot models that would allow us to focus the optimization effort on a low-dimensional portion of the system. We chose to exploit the local input-output linearizability of the actuators associated with knees and hips for example and put into the "weakly-actuated category" things like the global orientation of the body and possibly ankle joints. This allowed us to build trajectories of the full-model parameterized by initial conditions of a small subsystem, without making any approximations. In the end, we do build the control law for the full model around a low-dimensional model, just as advocates of pendulum models do, however, and this is important, all of the solutions of our low-dimensional model are feasible solutions of the robot itself and they meet whatever constraints were included in the design of the trajectories.

7.3 Original HZD vs G-HZD

Once one understands how the G-HZD tool works, it's hard to believe how much the previous method could accomplish with a single optimization. The work presented in [125] uses a single optimization to design the periodic orbit. And that is it. For robots with one degree of underactuation and for which a "mechanical phase variable" can be found, that is a strictly monotonically increasing generalized coordinate along the periodic orbit, the basic HZD result in [125] shows how to build an invariant surface, relate stability of the periodic orbit in the surface to a physical property of the periodic orbit[1], and how to render the surface exponentially attractive. G-HZD can

[1]The velocity of the robots center of mass should point downward at the end of the step.

handle more than one degree of underactuation. As in [100, 101, 65], G-HZD includes "time" as a monotonically increasing generalized coordinate. What makes it quite distinct from these references is that the equivalent of "G-HZD virtual constraints", the function $\nu(t, x_1)$, depends on time and the full state of the zero dynamics, thereby enriching the set of possible closed-loop behaviors.

With the previous work on HZD and its extensions in [20, 51, 48], we were unable to handle challenging terrain such as the Michigan Wave Field . This motivated the introduction of a family of periodic orbits in [32] and a first attempt at including transitions among the periodic orbits in [33] as a means of building in stability. While this latter thesis also used Supervised Machine Learning to design a feedback function, it also made some false steps relying on analogies with model predictive control that were not supported by deeper analysis. The present thesis is our attempt to provide a consistent design framework. An attractive feature of the original HZD approach is that it has an easily verifiable set of sufficient conditions for its many results. We hope in some future paper, a similarly clean analytical framework will be developed for G-HZD.

7.4 Stability Mechanism

Not only do the new results handle higher degrees of underactuation but even in the case of one degree of under actuation, the way stability is achieved is quite different in G-HZD vs. HZD. As discussed in the Introduction, with G-HZD, one does not have to count on the impacts to create stability. More general stability mechanisms, such as foot placement, or as shown in Figure 6.22, lowering the center of mass, naturally arise.

7.5 Future Work

We see this thesis as a first cut in developing a happy marriage among trajectory optimization, machine learning, and geometric nonlinear control. We hope the results in the thesis can be reinforced with easy-to-check sufficient conditions for our many assumptions. Beyond these technical considerations, we also see several other directions. The recent work in [28] may provide a geometric formulation for building the invariant surface, which would also clarify the choice of coordinates for making the mapping Ψ in (6.19) and its projection to be full rank. We believe the feedback linearization assumption on the "strongly actuated" part of the dynamics can be weakened considerably. Replacing the terminal condition in the optimization with a terminal penalty is another direction that needs to be investigated.

In a broader picture, a general motivation of legged robots is to travel through rough terrain. However, on flat ground and roadways, wheeled vehicles are obviously more energy efficient than legged robots. On the other hand, humanoid robots are specially made to mimic human motions so that they can work in human infrastructure and use tools designed for humans. Letting Cassie drive a Segway (Figure 7.1) conceptually merges the two ideas in one experiment, which is challenging in theory and experiment, and fun to watch. Based on preliminary bench tests on Cassie, it should be able to drive the Segway at 10 miles per hour (4.5 m/s), faster than the robot's maximum possible running speed. Cassie should also be able to drive the Segway on both longitudinal and lateral slopes to show a reliable driving skill. Lastly, Cassie should be able to make the Segway turn by shifting its knee sideways to push the knee support. The preliminary result is shown in Extension 5

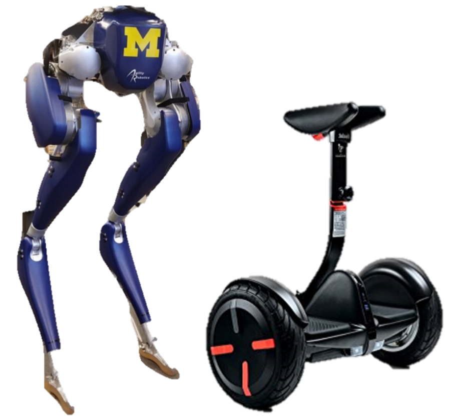


Figure 7.1: Cassie with a Segway. Cassie should drive the Segway miniPro as a human to move forward and backward, over flat ground and slopes, make turns and stop.

APPENDICES

APPENDIX A

Index to Multimedia Extensions

Table A.1: Table of Multimedia

Extension	Type	Description
1	Code	Inverted pendulum example
2	Video	2D Design to 3D Implementation
3	Video	Long Walk
4	Video	3D robot experiments
5	Video	Cassie on a Segway

APPENDIX B

Normal Forms for Mechanical Models

Consider a standard mechanical model

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = Bu$$

and let

$$\Omega(q,\dot{q}) := C(q,\dot{q})\dot{q} + G(q).$$

We suppose the system is *underactuated*, that is, there are fewer independent actuators than generalized coordinates. In fact, we suppose there exists a partition of the coordinates in which the model takes the form

$$\begin{aligned} D_{11}(q)\ddot{q}_1 + D_{12}(q)\ddot{q}_2 + \Omega_1(q,\dot{q}) &= 0 \\ D_{21}(q)\ddot{q}_1 + D_{22}(q)\ddot{q}_2 + \Omega_2(q,\dot{q}) &= B_2 u, \end{aligned} \tag{B.1}$$

with B_2 square and invertible. Because $D(q)$ is positive definite, by the Shur Complement Lemma, $D_{11}(q)$, $D_{22}(q)$, and

$$\bar{D}(q) := D_{22}(q) - D_{21}(q)D_{11}^{-1}(q)D_{12}(q) \tag{B.2}$$

are all positive definite as well.

Following [113], define

$$\begin{aligned} J^{\text{norm}}(q) &:= D_{11}^{-1}(q)D_{12}(q) \\ \bar{\Omega}_1(q,\dot{q}) &:= -D_{11}^{-1}(q)\Omega_1(q,\dot{q}) \\ \bar{\Omega}_2(q,\dot{q}) &:= \Omega_2(q,\dot{q}) - D_{21}(q)D_{11}^{-1}(q)\Omega_1(q,\dot{q}), \end{aligned} \tag{B.3}$$

Then the (regular) feedback

$$u = B_2^{-1}(q)\left(\bar{D}(q)v + \bar{\Omega}_2(q,\dot{q})\right), \tag{B.4}$$

results in the Spong normal form:

$$\begin{aligned} \ddot{q}_1 &= \bar{\Omega}_1(q,\dot{q}) - J^{\text{norm}}(q)v, \\ \ddot{q}_2 &= v. \end{aligned} \tag{B.5}$$

Defining $x_1 = (q_1, \dot{q}_1)$, $x_{2a} = q_2$, and $\dot{x}_{2b} = \dot{q}_2$, it follows that the model can be expressed as

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, v) \\ \dot{x}_{2a} &= x_{2b} \\ \dot{x}_{2b} &= v. \end{aligned} \tag{B.6}$$

An alternative form is developed in [125, pp. 62], which uses the conjugate momenta that arises from Lagrange's equations. It has the advantage that the input only appears in the second row of the model. Define

$$\sigma_1 := D_{11}(q)\dot{q}_1 + D_{12}(q)\dot{q}_2 \tag{B.7}$$

$$\dot{D}(q,\dot{q}) := \frac{d}{dt}D(q) \tag{B.8}$$

Then the model can also be written as

$$\begin{aligned}\dot{q}_1 &= D_{11}^{-1}(q)\left[\sigma_1 - D_{12}(q)\dot{q}_2\right]\\ \dot{\sigma}_1 &= \kappa_1(q,\sigma_1,\dot{q}_2)\\ \ddot{q}_2 &= v\end{aligned} \tag{B.9}$$

where

$$\begin{aligned}\kappa_1(q,\sigma_1,\dot{q}_2) := \Big(\dot{D}_{11}(q,\dot{q})\dot{q}_1 + \dot{D}_{12}(q,\dot{q})\dot{q}_2 -&\\ \Omega_1(q,\dot{q})\Big)\Big|_{\dot{q}_1 = D_{11}^{-1}(q)[\sigma_1 - D_{12}(q)\dot{q}_2]}&.\end{aligned} \tag{B.10}$$

With x_2 defined as above and $x_1 := (q_1, \sigma_1)$, the model takes the form

$$\begin{aligned}\dot{x}_1 &= f_1(x_1, x_2)\\ \dot{x}_{2a} &= x_{2b}\\ \dot{x}_{2b} &= v.\end{aligned} \tag{B.11}$$

Various authors prefer one of (B.6) and (B.11) to the other; both are useful.

APPENDIX C

Proofs

C.1 Proof of Proposition VI.1

The proof is most easily done using the method of Poincaré sections [89]. By A-1 and the assumption on u^{cu}, the closed-loop system (6.10) has period $T_p > 0$ and the origin is an equilibrium point. Due to the time-varying nature of the closed-loop system, we make time a state variable, and because the system is T_p-periodic, we add in a time-based reset map

$$\begin{cases} \quad \dot{\tau} = 1, & \tau^- < T_p \\ \quad \dot{x} = f^{cu}(\tau, x) := f(\tau, x, u^{cu}(\tau, x)), & \\ \tau^+ = 0 & \tau^- = T_p \\ x^+ = x^-. & \end{cases} \tag{C.1}$$

The notation τ^-, τ^+, x^- and x^+ is explained in Section 6.3. Because the state reset map is trivial, namely $x^+ = x^-$, the solutions of (6.10) and (C.1) are identical. Define a Poincaré section by

$$\mathcal{S}_n := \{(\tau, x) \in \mathbb{R}^{n+1} \mid \tau = T_p, x \in B\}, \tag{C.2}$$

which is an n-dimensional hypersurface in the state space of the model. Then, by construction of the closed-loop system, for $\xi \in \mathcal{S}_n$, the Poincaré map $P : \mathcal{S}_n \to \mathcal{S}_n$ is given by

$$P(\xi) = \varphi_\xi(T_p). \tag{C.3}$$

Indeed, for $\xi \in B$,

$$\begin{aligned}\varphi_\xi(t) &= \xi + \int_0^t f(\varphi_\xi(s), u_\xi(s))ds \\ &= \xi + \int_0^t f(\varphi_\xi(s), u^{cu}(s, \varphi_\xi(s))ds,\end{aligned}$$

due to (6.9). By A-4, $\xi^* = 0$ is a fixed point of the Poincaré map. Also by A-4, P is a contraction because for each $\xi \in B$, $V \circ P(\xi) \leq cV(\xi)$, and hence by induction,

$$V \circ P^k(\xi) \leq c^k V(\xi),$$

and by A-3,

$$||P^k(\xi)||^2 \leq c^k \frac{\alpha_2}{\alpha_1} ||\xi||^2 \xrightarrow[k\to\infty]{} 0,$$

proving local exponential stability of the fixed point. The uniformity in t_0 follows from periodicity. ■

C.2 Proof of Proposition VI.3

Without loss generality, we assume that B_{1cu} is bounded so that its closure is compact. Then there exists L, a Lipschitz constant, such that

$$||\gamma(x_1)||_2 \leq L||x_1||_2$$

for all $x_1 \in B_{1cu}$. Define $V_1 : B_1 \to \mathbb{R}$ by $V_1(x_1) := V(x_1, \gamma(x_1))$. It follows that

$$\alpha_1 x_1^\top x_1 \leq V_1(x_1) \leq \alpha_2(1 + L^2) x_1^\top x_1,$$

and hence V_1 is positive definite, with quadratic lower and upper bounds. From (6.15),

$$V_1(\varphi_{1\xi_1}(T_p)) \leq c V_1(\xi_1).$$

From here, the proof of Prop. VI.1 can be repeated and the result follows. ■

C.3 Proof of Theorem. VI.5

From the hypotheses of the Theorem and Prop. VI.3, the closed-loop system (6.25) is a cascade of two locally exponentially systems, namely, the second row of (6.25) and the reduced-order system (6.17). By standard results, the overall system is locally exponentially stable. By [118, Thm. 43, Section 5.1, pp. 143], because the system is periodically time-varying, the stability is uniform in t_0.

■

C.4 Proof of Corollary. VI.7

Defining y as in (6.24) results in the closed-loop system having the form

$$\begin{aligned} \dot{x}_1 &= f_1(t, x_1, \nu(t, x_1) + y, \mu(t, x_1) - [K_p \; K_d] y) \\ \dot{y} &= Ay, \end{aligned} \tag{C.4}$$

with A Hurwitz. Hence, the proof of Theorem VI.5 can be repeated and we are done.

■

C.5 Proof of Proposition VI.11

The proof is very similar to that of Prop. VI.1. Define a Poincaré section by

$$\mathcal{S}_n := \{(\tau, x) \in \mathbb{R}^{n+1} \mid \tau = T_p, x \in B\}, \tag{C.5}$$

which is an n-dimensional hypersurface in the state space of the model. Then, by construction of the closed-loop system, for $\xi \in \mathcal{S}_n$, the Poincaré map $P : \mathcal{S}_n \to \mathcal{S}_n$ is given by

$$P(\xi) = \varphi_\xi(T_p). \tag{C.6}$$

By H-4, $\xi^* := x_m^*$ is a fixed point of the Poincaré map. Also by H-4, P is a contraction because for each $\xi \in B$, $V \circ P(\xi) \leq cV(\xi)$, and hence by induction,

$$V \circ P^k(\xi) \leq c^k V(\xi),$$

and by H-3,

$$||P^k(\xi)||^2 \leq c^k \frac{\alpha_2}{\alpha_1} ||\xi - \xi^*||^2 \xrightarrow[k\to\infty]{} 0,$$

proving local exponential stability of the fixed point. Because the closed-loop system is locally Lipschitz continuous, local exponential stability of the fixed point implies exponential attractivity of the orbit

$$\mathcal{O} := \{(\tau, \varphi_{\xi^*}(\tau) \mid 0 \leq \tau < T_p\}.$$

Because $\tau(t) = t$, we have local exponential stability of the periodic solution. ■

C.6 Proof of Proposition VI.12

Without loss generality, we assume that B_{1cu} is bounded so that its closure is compact. Then there exists L, a Lipschitz constant, such that

$$||\gamma(x_1 - x_1^*)||_2 \leq L||x_1 - x_1^*||_2$$

for all $x_1 \in B_{1cu}$. Define $V_1 : B_1 \to \mathbb{R}$ by $V_1(x_1) := V(x_1, \gamma(x_1))$. It follows that

$$\begin{aligned} \alpha_1(x_1 - x_1^*)^\top(x_1 - x_1^*) \leq V_1(x_1) \leq \\ \alpha_2(1 + L^2)(x_1 - x_1^*)^\top(x_1 - x_1^*), \end{aligned}$$

and hence V_1 is positive definite, with quadratic lower and upper bounds. From (6.15),

$$V_1(\varphi_{1\xi_1}(T_p)) \leq cV_1(\xi_1).$$

From here, the proof of Prop. VI.11 can be repeated and the result follows. ■

C.7 Proof of Theorem. VI.14

The Poincaré section is defined as in (C.5). References [50] and [125, Chap. 4] show how to reduce the stability analysis of a hybrid model with two continuous phases to that of an equivalent hybrid system with a single continuous phase. We build the equivalent hybrid system with the continuous phase from Σ_{ii} and a reset map Δ_{eq} that captures the flow of Σ_{i}, viz

$$\begin{cases} \dot{\tau} = 1 & \tau^- < T_p \\ \dot{x} = f(x_1, x_2, u_1, u_2) & \\ u_1 = \mu_{1ii}(\tau, x_1) & \\ u_2 = \mu_{2ii}(\tau, x_1) - \begin{bmatrix} \frac{K_p}{\epsilon^2} & \frac{K_d}{\epsilon} \end{bmatrix} \big(x_2 - \nu_{ii}(\tau, x_1)\big) & \\ \begin{bmatrix} \tau \\ x \end{bmatrix}^+ = \Delta_{eq}(\tau^-, x^-), & \tau^- = T_p \end{cases} \tag{C.7}$$

With this construction and Prop. VI.12, the zero dynamics manifold is

$$\mathcal{Z} := \{(\tau, x_1, x_2) \mid x_2 = \nu_{ii}(\tau, x_1)\},$$

and the restricted Poincaré map $\rho : \mathcal{S}_e \cap \mathcal{Z} \to \mathcal{S}_e \cap \mathcal{Z}$ has x_1^* as a locally exponentially stable fixed point. The equivalent hybrid system (C.7) therefore satisfies all the hypotheses of [4, Thm. 2], and hence the periodic orbit

$$\mathcal{O} := \{(\tau, \varphi_{\xi_1^*}(\tau) \mid 0 \leq \tau < T_p\}$$

is locally exponentially stable. ■

APPENDIX D

Relation to Backstepping, Zero Dynamics, and Immersion and Invariance

For definiteness, consider a standard Lagrangian dynamical model where $q \in \mathbb{R}^n$ is a set of generalized coordinates and $u \in \mathbb{R}^m$ is a vector of torques,

$$D(q)\ddot{q} + H(q, \dot{q}) = B(q). \tag{D.1}$$

Assume the system is underactuated, that is, $n > m$, and that the coordinates have been decomposed as

$$q := \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}, \tag{D.2}$$

in which the model takes the form

$$\begin{aligned} D_{11}(q)\ddot{q}_1 + D_{12}(q)\ddot{q}_2 + H_1(q, \dot{q}) &= 0 \\ D_{21}(q)\ddot{q}_1 + D_{22}(q)\ddot{q}_2 + H_2(q, \dot{q}) &= B_2(q)u. \end{aligned} \tag{D.3}$$

with $B_2(q)$ square and full rank. References [114, 103, 126] show that there is a regular feedback that places the system in the form

$$\begin{aligned}\dot{x}_1 &= f_1(x_1, q_2, \dot{q}_2)\\ \ddot{q}_2 &= v,\end{aligned} \tag{D.4}$$

with

$$x_1 := \begin{bmatrix} q_1 \\ \sigma_1 \end{bmatrix} \text{ and } \sigma_1 := D_{11}(q)\dot{q}_1 + D_{12}(q)\dot{q}_2,$$

the generalized momentum conjugate to q_1.

D.1 Backstepping

To begin the backstepping process in (D.4), one needs a feedback

$$\begin{bmatrix} q_2 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} \nu_a(x_1) \\ \nu_b(x_1) \end{bmatrix}$$

that renders the origin of the reduced-order system

$$\dot{x}_1 = f_1(x_1, \nu_a(x_1), \nu_b(x_1)) \tag{D.5}$$

locally exponentially stable with a known Lyapunov function. However, to pull this feedback through the double integrator, it must be true that $\nu_b(x_1) = \frac{d}{dt}\nu_a(x_1)$, that is

$$\nu_b(x_1) = \left[\frac{\partial}{\partial x_1}\nu_a(x_1)\right] f_1(x_1, \nu_a(x_1), \nu_b(x_1)). \tag{D.6}$$

Backstepping does not provide any systematic means to meet the required integrability condition. Of course, if the system has the form

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, q_2) \\ \ddot{q}_2 &= v, \end{aligned} \tag{D.7}$$

then there is no integrability constraint and backstepping can be done, assuming one is clever enough to find a feedback $q_2 = \nu_a(x_1)$ that renders the origin of

$$\dot{x_1} = f_1(x_1, \nu_a(x_1))$$

locally exponentially stable. The solution we presented in Sect. 6.1.4 uses trajectory optimization to automatically build a feedback that satisfies the integrability condition (D.6) and provides for local exponential stability. Moreover, bounds on inputs and other constraints can potentially be included in the trajectory optimization process, whereas they are challenging to incorporate into backstepping.

D.2 Zero Dynamics

The method of Hybrid Zero Dynamics as developed in [126] exploits the structure of f_1 in (D.5), namely

$$\frac{d}{dt}\begin{bmatrix} q_1 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \dot{q}_1 \\ f_{1b}(q_1, \dot{q}_1, q_2, \dot{q}_2) \end{bmatrix}$$

and

$$\dot{q}_1 = D_{11}^{-1}(q)\left[\sigma_1 - D_{12}(q)\dot{q}_2\right],$$

to solve for a solution of the form

$$\begin{bmatrix} q_2 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} h_d(q_1) \\ \left(\frac{\partial}{\partial q_1} h_d(q_1)\right) \dot{q}_1 \end{bmatrix},$$

so that the integrability condition (D.6) is automatically met. When the computational method in [58, 53] does produce a solution, it does not come with a Lyapunov function and hence input-output linearization is often used to "pull" the virtual constraints back through the double integrators. Moreover, conditions for the *virtual constraints* $q_2 = h_d(q_1)$ to stabilize (a hybrid version of) (D.5) are only known when q_1 is a scalar. The solution we have given in Sect. 6.3 works for vector valued q_1, hence for models with more than one degree of underactuation. Moreover, even for one degree of underactuation, it provides a more general solution to the boundary value problem in that it naturally produces solutions of the form $h_d(t, q_1, \dot{q}_1)$, that is, the controller depends in a non-trivial way on the full state of the x_1-subsystem.

D.3 Immersion and Invariance

The method of immersion and invariance (I&I) presented in [11, 62, 119] is more general than backstepping and can provide alternative cascade realizations to the simple one used in (6.25). However, I&I still requires a target system to be provided, such as, (6.17), which is what our method is constructing. In other words, once a feedback satisfying Prop. VI.3 has been constructed, I&I can be used to build alternatives to the feedback used in Thm. VI.5, but it will not replace the design of the reduced-order model.

D.4 Standard MPC

Figure D.1 shows the standard MPC with Zero-Order-Hold (ZOH) condition,

$$u^{zoh}(t,\xi) = u_\xi(0), t \in [0, T_p). \tag{D.8}$$

We would like to implement this controller to the high dimensional system. However, it does not scale well. The learned feedback $\mu(t,x)$ shows the similar performance as the classic MPC. Sect. 6.1.3 illustrates how to apply it to a reduced-order model

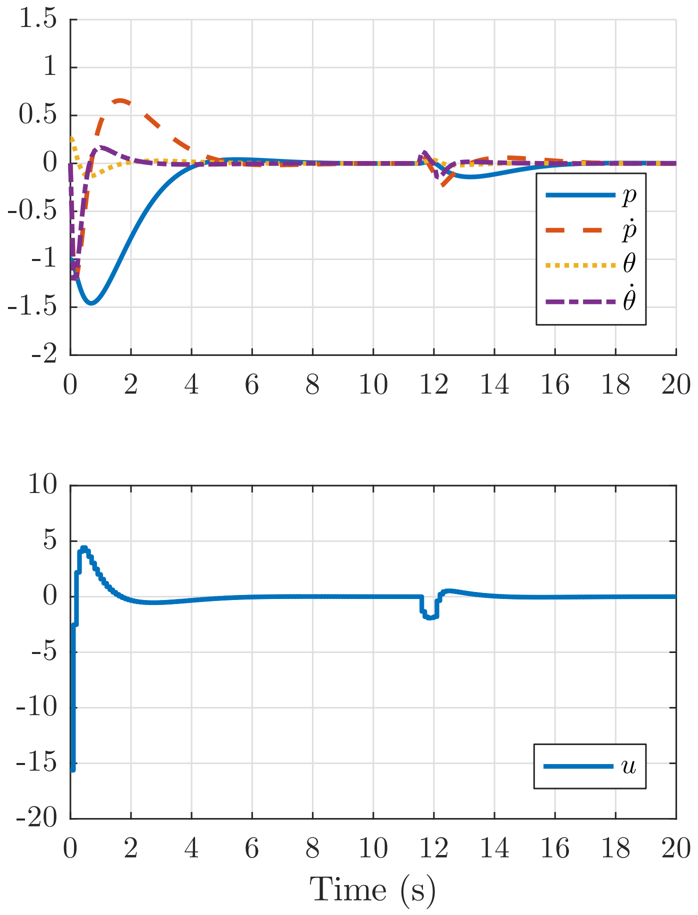


Figure D.1: A classic MPC controller is applied to the same system as in Figure 6.8

while embedding it to the full-order model. In this sense, the curse of dimensionality has been mitigated.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in neural information processing systems*, pages 1–8, 2007.

[2] N. Aghasadeghi, H. Zhao, L. J. Hargrove, A. D. Ames, E. J. Perreault, and T. Bretl. Learning impedance controller parameters for lower-limb prostheses. In *Intelligent robots and systems (IROS), 2013 IEEE/RSJ international conference on*, pages 4268–4274. IEEE, 2013.

[3] A. Agrawal, O. Harib, A. Hereid, S. Finet, M. Masselin, L. Praly, A. D. Ames, K. Sreenath, and J. W. Grizzle. First steps towards translating HZD control of bipedal robots to decentralized control of exoskeletons. *IEEE Access*, 5:9919–9934, 2017.

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

[5] A. D. Ames. Human-inspired control of bipedal walking robots. *IEEE Transactions on Automatic Control*, 59(5):1115–1130, May 2014.

[6] A. D. Ames. Human-inspired control of bipedal walking robots. *Automatic Control, IEEE Transactions on*, 59(5):1115–1130, 2014.

[7] A. D. Ames, E. A. Cousineau, and M. J. Powell. Dynamically stable bipedal robotic walking with NAO via human-inspired hybrid zero dynamics. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, pages 135–144. ACM, 2012.

[8] A. D. Ames, E. A. Cousineau, and M. J. Powell. Dynamically stable robotic walking with NAO via human-inspired hybrid zero dynamics. In *Hybrid Systems, Computation and Control (HSCC)*, Philadelphia, April 2012.

[9] A. D. Ames, R. D. Gregg, and M. W. Spong. A geometric approach to three-dimensional hipped bipedal robotic walking. In *45th Conference on Decision and Control*, San Diego, CA, 2007.

[10] S. Apostolopoulos, M. Leibold, and M. Buss. Settling time reduction for underactuated walking robots. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 6402–6408. IEEE, 2015.

[11] A. Astolfi and R. Ortega. Immersion and invariance: a new tool for stabilization and adaptive control of nonlinear systems. *IEEE Transactions on Automatic Control*, 48(4):590 – 606, April 2003.

[12] J.-P. Aubin, J. Lygeros, M. Quincampoix, S. Sastry, and N. Seube. Impulse differential inclusions: a viability approach to hybrid systems. *IEEE Transactions on Automatic Control*, 47(1):2–20, 2002.

[13] C. Azevedo, P. Poignet, and B. Espiau. Moving horizon control for biped robots without reference trajectory. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 3, pages 2762–2767, 2002.

[14] C. Azevedo, P. Poignet, and B. Espiau. Artificial locomotion control: from human to robots. *Robotics and Autonomous Systems*, 47(4):203–223, 2004.

[15] D. Bainov and P. Simeonov. *Systems with Impulse Effects : Stability, Theory and Applications*. Ellis Horwood Limited, Chichester, 1989.

[16] P. A. Bhounsule, J. Cortell, A. Grewal, B. Hendriksen, J. D. Karssen, C. Paul, and A. Ruina. 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.

[17] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316 [cs.CV]*, 2016.

[18] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[19] Boston Dynamics. Atlas, The Next Generation. *https://www.youtube.com/watch?v=rVlhMGQgDkY*, April 2016.

[20] B. G. Buss, K. A. Hamed, B. A. Griffin, and J. W. Grizzle. Experimental results for 3D bipedal robot walking based on systematic optimization of virtual constraints. In *American control conference*, 2016.

[21] B. G. Buss, K. A. Hamed, B. A. Griffin, and J. W. Grizzle. Experimental results for 3D bipedal robot walking based on systematic optimization of virtual constraints. In *submitted to American Control Conference*, Boston, MA, June 2016.

[22] B. G. Buss, A. Ramezani, K. Akbari Hamed, B. A. Griffin, K. S. Galloway, and J. W. Grizzle. Preliminary walking experiments with underactuated 3D bipedal robot marlo. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2529–2536. IEEE, 2014.

[23] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas, and J. W. Grizzle. RABBIT: a testbed for advanced control theory. *IEEE Control Systems Magazine*, 23(5):57–79, Oct. 2003.

[24] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas-De-Wit, and J. W. Grizzle. Rabbit: a testbed for advanced control theory. *IEEE Control Systems*, 23(5):57–79, Oct 2003.

[25] F. H. Clarke, Y. S. Ledyaev, E. D. Sontag, and A. I. Subbotin. Asymptotic controllability implies feedback stabilization. *IEEE Transactions on Automatic Control*, 42(10):1394–1407, 1997.

[26] S. H. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–85, 2005.

[27] S. H. Collins, M. Wisse, and A. Ruina. A three-dimensional passive-dynamic walking robot with two legs and knees. *International Journal of Robotics Research*, 20(7):607–615, July 2001.

[28] L. Consolini, A. Costalunga, and M. Maggiore. A coordinate-free theory of virtual holonomic constraints. *arXiv preprint arXiv:1709.07726 [math.OC]*, 2016.

[29] L. Consolini, A. Costalunga, and M. Maggiore. Synthesis of safe controller via supervised learning for truck lateral control. *arXiv preprint arXiv:1712.05506 [cs.SY]]*, 2018.

[30] J.-M. Coron. On the stabilization of controllable and observable systems by an output feedback law. *Mathematics of Control, Signals, and Systems (MCSS)*, 7(3):187–216, 1994.

[31] J.-M. Coron, L. Praly, and A. Teel. Feedback stabilization of nonlinear systems: Sufficient conditions and lyapunov and input-output techniques. In *Trends in control*, pages 293–348. Springer, 1995.

[32] X. Da, O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle. From 2D design of underactuated bipedal gaits to 3D implementation: Walking with speed tracking. *IEEE Access*, 4:3469–3478, 2016.

[33] X. Da, R. Hartley, and J. W. Grizzle. Supervised learning for stabilizing underactuated bipedal robot locomotion, with outdoor experiments on the wave field. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3476–3483, May 2017.

[34] H. Demuth and M. Beale. *Neural network toolbox for use with MATLAB*. Citeseer, 1993.

[35] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[36] E. R. Dunn and R. D. Howe. Foot placement and velocity control in smooth bipedal walking. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 578–583. IEEE, 1996.

[37] Dynamic Robotics Laboratory. ATRIAS: An Agile and Efficient Bipedal Robot. *https://www.youtube.com/watch?v=YFEJvb8iM7A*, April 2015.

[38] Dynamic Robotics Laboratory. ATRIAS Robot: First 3D Test. *https://www.youtube.com/watch?v=vq4Xq4eSCv8*, Feb 2015.

[39] K. R. Embry, D. J. Villarreal, and R. D. Gregg. A unified parameterization of human gait across ambulation modes. In *Submit to: International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2016.

[40] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, and E. Todorov. An integrated system for real-time model predictive control of humanoid robots. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 292–299, Oct 2013.

[41] S. Faraji, S. Pouya, C. G. Atkeson, and A. J. Ijspeert. Versatile and robust 3D walking with a simulated humanoid robot (Atlas): a model predictive control approach. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1943–1950. IEEE, 2014.

[42] R. Full and D. Koditschek. Templates and anchors: Neuromechanical hypotheses of legged locomotion on land. *Journal of Experimental Biology*, 202:3325–3332, December 1999.

[43] A. Goswami, B. Espiau, and A. Keramane. Limit cycles in a passive compass gait biped and passivity-mimicking control laws. *Autonomous Robots*, 4(3):273–86, 1997.

[44] R. D. Gregg. Controlled reduction of a five-link 3D biped with unactuated yaw. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 669–674. IEEE, 2011.

[45] R. D. Gregg, T. Lenzi, L. J. Hargrove, and J. W. Sensinger. Virtual constraint control of a powered prosthetic leg: From simulation to experiments with transfemoral amputees. *IEEE Transactions on Robotics*, 2014.

[46] R. D. Gregg and L. Righetti. Controlled reduction with unactuated cyclic variables: Application to 3d bipedal walking with passive yaw rotation. *Automatic Control, IEEE Transactions on*, 58(10):2679–2685, 2013.

[47] R. D. Gregg, E. J. Rouse, L. J. Hargrove, and J. W. Sensinger. Evidence for a time-invariant phase variable in human ankle control. *PloS one*, 9(2):e89163, 2014.

[48] B. Griffin and J. Grizzle. Nonholonomic virtual constraints and gait optimization for robust walking control. *The International Journal of Robotics Research*, page 0278364917708249, 2016.

[49] J. Grizzle, J. Hurst, B. Morris, H.-W. Park, and K. Sreenath. MABEL, a new robotic bipedal walker and runner. In *American Control Conference, 2009. ACC '09.*, pages 2030–2036, 2009.

[50] J. W. Grizzle, C. Chevallereau, R. W. Sinnet, and A. D. Ames. Models, feedback control, and open problems of 3D bipedal robotic walking. *Automatica*, 50(8):1955–1988, 2014.

[51] K. A. Hamed, B. G. Buss, and J. W. Grizzle. Exponentially stabilizing continuous-time controllers for periodic orbits of hybrid systems: Application to bipedal locomotion with ground height variations. *The International Journal of Robotics Research*, 35(8):977–999, 2016.

[52] O. Harib, A. Hereid, A. Agrawal, T. Gurriet, S. Finet, G. Boeris, A. Duburcq, M. E. Mungai, M. Masselin, A. D. Ames, K. Sreenath, and J. Grizzle. Feedback control of an exoskeleton for paraplegics: Toward robustly stable hands-free dynamic walking. *arXiv preprint arXiv:1802.08322 [cs.RO]*, 2018.

[53] A. Hereid, E. A. Cousineau, C. M. Hubicki, and A. D. Ames. 3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[54] A. Hereid, S. Kolathaya, and A. D. Ames. Online hybrid zero dynamics optimal gait generation using legendre pseudospectral optimization. In *To appear in: IEEE Conference on Decision and Control (CDC)*. IEEE, 2016.

[55] A. Hereid, S. Kolathaya, M. S. Jones, J. Van Why, J. W. Hurst, and A. D. Ames. Dynamic multi-domain bipedal walking with ATRIAS through SLIP based human-inspired control. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pages 263–272, New York, NY, USA, 2014. ACM.

[56] K. Hosoda, T. Takuma, and M. Ishikawa. Design and control of a 3D biped robot actuated by antagonistic pairs of pneumatic muscles. In *Proceedings of International Symposium on Adaptive Motion in Animals and Machines*, 2005.

[57] A. Isidori. *Nonlinear Control Systems.* Springer-Verlag, Berlin, third edition, 1995.

[58] M. S. Jones. Optimal control of an underactuated bipedal robot. Master's thesis, Oregon State University, ScholarsArchive@OSU, June 2014.

[59] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-theory and basic experiment. In *Advanced Robotics, 1991.'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 741–746. IEEE, 1991.

[60] S. Kajita, T. Yamaura, and A. Kobayashi. Dynamic walking control of biped robot along a potential energy conserving orbit. *IEEE Transactions on Robotics and Automation*, 8(4):431–437, August 1992.

[61] K. Kaneko, F. Kanehiro, M. Morisawa, K. Akachi, G. Miyamori, A. Hayashi, and N. Kanehira. Humanoid robot HRP-4 – humanoid robotics platform with lightweight and slim body. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4400–4407. IEEE, 2011.

[62] D. Karagiannis, A. Astolfi, and R. Ortega. Nonlinear stabilization via system immersion and manifold invariance: survey and new results. *Multiscale Modeling & Simulation*, 3(4):801–817, 2005.

[63] H. K. Khalil. *Nonlinear Systems.* Prentice Hall, Upper Saddle River, NJ, third edition, 2002.

[64] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard. Whole-body model-predictive control applied to the HRP-2 humanoid. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3346–3351. IEEE, 2015.

[65] S. Kolathaya and A. D. Ames. Parameter to state stability of control lyapunov functions for hybrid system models of robots. *Nonlinear Analysis: Hybrid Systems*, 25:174–191, 2017.

[66] F. H. Kong, A. M. Boudali, and I. R. Manchester. Phase-indexed ilc for control of underactuated walking robots. In *Control Applications (CCA), 2015 IEEE Conference on*, pages 1467–1472. IEEE, 2015.

[67] T. Koolen, T. de Boer, J. Rebula, A. Goswami, and J. Pratt. Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models. *The International Journal of Robotics Research*, 31(9):1094–1113, July 2012.

[68] M. Krause, J. Englsberger, P.-B. Wieber, and C. Ott. Stabilization of the capture point dynamics for bipedal walking based on model predictive control. *IFAC Proceedings Volumes*, 45(22):165–171, 2012.

[69] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016.

[70] A. D. Kuo. Stabilization of lateral motion in passive dynamic walking. *International Journal of Robotics Research*, 18(9):917–930, 1999.

[71] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[72] C. Liu, C. G. Atkeson, and J. Su. Biped walking control using a trajectory library. *Robotica*, 31(02):311–322, 2013.

[73] I. R. Manchester and J. Umenberger. Real-time planning with primitives for dynamic walking over uneven terrain. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4639–4646. IEEE, 2014.

[74] A. E. Martin, D. C. Post, and J. P. Schmiedeler. Design and experimental implementation of a hybrid zero dynamics-based controller for planar bipeds with curved feet. *The International Journal of Robotics Research*, 33(7):988–1005, 2014.

[75] A. E. Martin, D. C. Post, and J. P. Schmiedeler. Design and experimental implementation of a hybrid zero dynamics-based controller for planar bipeds with curved feet. *The International Journal of Robotics Research*, 33(7):988–1005, 2014.

[76] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[77] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9(2):62–82, Apr. 1990.

[78] T. McGeer. Passive walking with knees. In *Proc. of the 1990 IEEE International Conference on Robotics and Automation, Cincinnati, OH*, volume 3, pages 1640–1645, May 1990.

[79] I. Mordatch, K. Lowrey, and E. Todorov. Ensemble-CIO: Full-body dynamic motion planning that transfers to physical humanoids. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 5307–5314. IEEE, 2015.

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

[81] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer, 2006.

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

[83] Q. Nguyen, X. Da, J. W. Grizzle, and K. Sreenath. Dynamic walking on stepping stones with gait library and control barrier. In *Workshop on Algorithimic Foundations of Robotics (WAFR)*, 2016.

[84] Y. Ogura, H. Aikawa, K. Shimomura, A. Morishima, H.-o. Lim, and A. Takanishi. Development of a new humanoid robot WABIAN-2. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 76–81. IEEE, 2006.

[85] T. Parisini and R. Zoppoli. A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica*, 31(10):1443–1451, 1995.

[86] H. Park, A. Ramezani, and J. W. Grizzle. A finite-state machine for accommodating unexpected large ground height variations in bipedal robot walking. *IEEE Transactions on Robotics*, 29(29):331–345, 2013.

[87] H.-W. Park, K. Sreenath, J. W. Hurst, and J. W. Grizzle. Identification of a bipedal robot with a compliant drivetrain. *IEEE Control Systems Magazine*, 31(2):63 –88, april 2011.

[88] I. Park, J. Kim, J. Lee, and J. Oh. Mechanical design of humanoid robot platform KHR-3 (KAIST Humanoid Robot 3: HUBO). In *Proc. 5th IEEE—RAS Int. Conf. Humanoid Robots*, pages 321–326, 2005.

[89] T. S. Parker and L. O. Chua. *Practical Numerical Algorithms for Chaotic Systems.* Springer-Verlag, New York, NY, 1989.

[90] F. Pfeiffer, K. Loffler, and M. Gienger. The concept of jogging Johnnie. In *IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002.

[91] M. J. Powell, A. Hereid, and A. D. Ames. Speed regulation in 3D robotic walking through motion transitions between human-inspired partial hybrid zero dynamics. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4803–4810. IEEE, 2013.

[92] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture Point: A Step toward Humanoid Push Recovery. *2006 6th IEEE-RAS International Conference on Humanoid Robots*, 2006.

[93] J. Pratt, T. Koolen, T. de Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus. Capturability-based analysis and control of legged locomotion, Part 2: Application to M2V2, a lower-body humanoid. *The International Journal of Robotics Research*, 31(10):1117–1133, Aug. 2012.

[94] J. Pratt and R. Tedrake. Velocity-based stability margins for fast bipedal walking. In M. Diehl and K. Mombaur, editors, *Fast Motions in Biomechanics and Robotics*, volume 340 of *Lecture Notes in Control and Information Sciences*, pages 299–324. Springer Berlin Heidelberg, 2006.

[95] M. H. Raibert. Legged robots. *Communications of the ACM*, 29(6):499–514, 1986.

[96] M. H. Raibert. *Legged robots that balance.* MIT Press, Mass., 1986.

[97] A. Ramezani, J. W. Hurst, K. Akbari Hamed, and J. W. Grizzle. Performance Analysis and Feedback Control of ATRIAS, A Three-Dimensional Bipedal Robot. *Journal of Dynamic Systems, Measurement, and Control*, 136(2), 2014.

[98] H. Razavi, A. M. Bloch, C. Chevallereau, and J. W. Grizzle. Symmetry in 3D legged locomotion: A new method for designing stable periodic gaits. *Autonomous Robots*, 2017.

[99] H. Razavi, A. M. Bloch, C. Christine, and J. W. Grizzle. Restricted discrete invariance and self-synchronization for stable walking of bipedal robots. In *Proceedings of the American Control Conference*, July 2015.

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

[101] J. P. Reher, A. Hereid, S. Kolathaya, C. M. Hubicki, and A. D. Ames. Algorithmic foundations of realizing multi-contact locomotion on the humanoid robot DURUS. In *The International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[102] D. Renjewski, A. Sprowitz, A. Peekema, M. Jones, and J. Hurst. Exciting engineered passive dynamics in a bipedal robot. *Robotics, IEEE Transactions on*, 31(5):1244–1251, 2015.

[103] M. Reyhanoglu, A. van der Schaft, N. McClamroch, and I. Kolmanovsky. Dynamics and control of a class of underactuated mechanical systems. *IEEE Transactions on Automatic Control*, 44(9):1663–1671, 1999.

[104] S. Rezazadeh, C. Hubicki, M. Jones, A. Peekema, J. Van Why, A. Abate, and J. W. Hurst. Spring-mass walking with ATRIAS in 3D: Robust gait control spanning zero to 4.3 kph on a heavily underactuated bipedal robot. *ASME Dynamic Systems and Control Conference (DSCC)*, page 23, 2015.

[105] C. O. Saglam and K. Byl. Meshing hybrid zero dynamics for rough terrain walking. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5718–5725. IEEE, 2015.

[106] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent ASIMO: system overview and integration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2478–83, Lausanne, Switzerland, 2002.

[107] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[108] B. Schürmann and M. Althoff. Convex interpolation control with formal guarantees for disturbed and constrained nonlinear systems. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 121–130. ACM, 2017.

[109] H. Shim and A. R. Teel. Asymptotic controllability and observability imply semiglobal practical asymptotic stabilizability by sampled-data output feedback. *Automatica*, 39(3):441–454, 2003.

[110] R. W. Sinnet and A. D. Ames. 2D bipedal walking with knees and feet: A hybrid control approach. In *48th IEEE Conference on Decision and Control*, Shanghai, P.R. China, 2009.

[111] R. W. Sinnet and A. D. Ames. 3D bipedal walking with knees and feet: A hybrid geometric approach. In *48th IEEE Conference on Decision and Control*, Shanghai, P.R. China, 2009.

[112] E. D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science & Business Media, 2013.

[113] M. W. Spong. Partial feedback linearization of underactuated mechanical systems. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Munich, Germany*, pages 314–321, September 1994.

[114] M. W. Spong. Energy based control of a class of underactuated mechanical systems. In *Proc. of IFAC World Congress, San Francisco, CA*, pages 431–435, 1996.

[115] K. Sreenath, H. Park, and J. W. Grizzle. Design and experimental implementation of a compliant hybrid zero dynamics controller with active force control for running on MABEL. In *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pages 51–56, 2012.

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

[117] K. Sreenath, H.-W. Park, I. Poulakakis, and J. Grizzle. Embedding active force control within the compliant hybrid zero dynamics to achieve stable, fast running on MABEL. *The International Journal of Robotics Research*, 32(3):324–345, 2013.

[118] M. Vidyasagar. *Nonlinear systems analysis*. SIAM, 2002.

[119] L. Wang, F. Forni, R. Ortega, Z. Liu, and H. Su. Immersion and invariance stabilization of nonlinear systems via virtual and horizontal contraction. *IEEE Transactions on Automatic Control*, 62(8):4017–4022, 2017.

[120] T. Wang and C. Chevallereau. Stability analysis and time-varying walking control for an under-actuated planar biped robot. *Robotics and Autonomous Systems*, 59(6):444 – 456, 2011.

[121] P. M. Wensing and D. Orin. 3D-slip steering for high-speed humanoid turns. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 4008–4013. IEEE, 2014.

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

[123] E. R. Westervelt, G. Buche, and J. W. Grizzle. Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds. *International Journal of Robotics Research*, 23(6):559–82, 2004.

[124] E. R. Westervelt, J. W. Grizzle, and C. Canudas. Switching and PI control of walking motions of planar biped walkers. *IEEE Transactions on Automatic Control*, 48(2):308–12, Feb. 2003.

[125] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. Choi, and B. Morris. *Feedback Control of Dynamic Bipedal Robot Locomotion*. Control and Automation. CRC Press, Boca Raton, FL, June 2007.

[126] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek. Zero dynamics of planar biped walkers with one degree of under actuation. In *IFAC World Congress*, Barcelona, Spain, July 2002.

[127] M. Wisse. Three additions to passive dynamic walking: actuation, an upper body, and 3D stability. *International Journal of Humanoid Robotics*, 2(04):459–478, 2005.

[128] M. Wisse, A. L. Schwab, and R. v. Linde. A 3D passive dynamic biped with yaw and roll compensation. *Robotica*, 19(03):275–284, 2001.

[129] A. Wu and H. Geyer. The 3-D spring–mass model reveals a time-based deadbeat control for highly robust running and steering in uncertain environments. *Robotics, IEEE Transactions on*, 29(5):1114–1124, 2013.

[130] J. Yamaguchi, E. Soga, S. Inoue, and A. Takanishi. Development of a bipedal humanoid robot-control method of whole body cooperative dynamic biped walking. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 368–374. IEEE, 1999.

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

[132] H. Zhao, J. Horn, J. Reher, V. Paredes, and A. D. Ames. A hybrid systems and optimization-based control approach to realizing multi-contact locomotion on transfemoral prostheses. In *IEEE Conference on Decision and Control (CDC)*, pages 1607–1612, Dec 2015.