Robust Machine Learning for the Control of Real-world Robotic Systems

Tyler Westenbroek

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-88
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-88.html

May 10, 2023

Robust Machine Learning for the Control of Real-world Robotic Systems

by

Tyler Westenbroek

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering — Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor S. Shankar Sastry, Chair
Professor Claire J. Tomlin
Professor Koushil Sreenath

Spring 2023

The dissertation of Tyler Westenbroek, titled Robust Machine Learning for the Control of Real-world Robotic Systems, is approved:

Chair _____ Date _____

_____ Date _____

_____ Date _____

University of California, Berkeley

Robust Machine Learning for the Control of Real-world Robotic Systems

Abstract

Robust Machine Learning for the Control of Real-world Robotic Systems

by

Tyler Westenbroek

Doctor of Philosophy in Engineering — Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor S. Shankar Sastry, Chair

Optimal control is a powerful paradigm for controller design as it can be used to implicitly encode complex stabilizing behaviors using cost functions which are relatively simple to specify. On the other hand, the curse of dimensionality and the presence of non-convex optimization landscapes can make it challenging to reliably obtain stabilizing controllers for complex high-dimensional systems. Recently, sampling-based reinforcement learning approaches have enabled roboticists to obtain approximately optimal feedback controllers for high-dimensional systems even when the dynamics are unknown. However, these methods remain too unreliable for practical deployment in many application domains.

This dissertation argues that the key to reliable optimization-based controller synthesis is obtaining a deeper understanding of how the cost functions we write down and the algorithms we design interact with the underlying feedback geometry of the control system. First, we next investigate how to accelerate model-free reinforcement learning by embedding control Lyapunov functions — which are energy like functions for the system— into the objective. Next we will introduce a novel data-driven policy optimization framework which embeds structural information from an approximate dynamics model and family of low-level feedback controllers into the update scheme. We then turn to a dynamic programming perspective, and investigate how the geometric structure of the system places fundamental limitations on how much computation is required to compute or learn a stabilizing controller. Finally, we investigate derivative-based search algorithms and investigate how to design 'good' cost functions for model predictive control schemes, which ensure these methods stabilize the system even when gradient-based methods are used to search over a non-convex objective. Throughout an emphasis will be placed on how structural insights gleaned from a simple analytical model can guide our design decisions, and we will discuss applications to dynamic walking, flight control, and autonomous driving.

Contents

List of Figures

List of Tables

Acknowledgments

To my family. Both the family I entered Berkeley with, and the family I pieced together along the way.

Chapter 1

Background and Motivation

As we strive to integrate autonomous robots into our daily lives, it is essential to control how these systems interact with dynamic, uncertain and difficult to model environments. Traditionally, the starting point for the engineering process has been simplified, often reduced-order dynamics models for the system. While these models do not capture every feature of the full-order system, they provide an intuitive interface through which engineers can reason about the real-world. By adopting systematic, analytical and hierarchical design techniques, engineering teams have been able to build up robust control strategies for complex systems such as walking robots using traditional model-based design techniques. A key benefit of these approaches is that they often provide intuitive knobs for engineers to turn, such a feedback gains with clear intuitive interpretations, enabling engineers to fine-tune the real-world behavior these approaches produce.

However, the sheer complexity that arises when shaping the real-world behavior of these systems presents daunting engineering challenges whose resolution is a prerequisite for real-world autonomy. Consider as an example the A1 quadruped which is used an example in Chapters 2 and 3. In order to produce successful real-world locomotion on this platform engineers must coordinate the motion of 18 underactuated degrees of freedom. Moreover, the walking controller must leverage intermittent contact with surfaces which are difficult to model and, given current perception limitations, difficult to even localize precisely with respect to the position of the robot. If an unintended event occurs, such as the slipping of a foot, the the controller must react within a fraction of a second to produce an entirely different sequence of motions. Put plainly, it is intractable for engineering teams to write down in detail the behavior that complex hardware platforms should display.

Moreover, even if we *could* write down the desired behavior it is exceedingly difficult to actually *execute* that behavior in the real world. Continuing with the previous example, when walking on compliant or rigid surfaces the forces the robot must exert on it surroundings to produce desired accelerations at the joint level can vary wildly. Moreover, the space of feasible system trajectories may be completely different, and it may not even be possible to track a planned reference motion. The rigid body models we typically use to design

controllers for our robots simply cannot handle this scope of variation, and it is impractical to model the characteristics of every possible environment our robots will be required to interact with in the real world. Thus, what use are our models if they fail us in situations where rapid adaptation is the most crucial?

Recently there has been optimism around using techniques from machine learning, such as deep reinforcement learning [35, 2, 60, 61], to tackle these grand challenges. These approaches have shown great promise in automatically generating complex behaviors for high-dimensional hardware platforms and, at least in principle, can learn to improve performance by learning directly from real-world data, eschewing the need for a structured representation of the system dynamics. The current state-of-the art in quadrupedal locomotion is dominated by 'sim-to-real' reinforcement learning approaches [77, 40, 39] which train walking controllers under a wide range of perturbations to the nominal dynamics model in simulation and then train the resulting controller in the real world.

However, despite a number of impressive demonstrations over the last few years, the real world performance of these methods leaves room for improvement and it is extremely difficult to modify the real-world behavior of these black box approaches. As stated above, one possibility is to learn directly from real world data, using a fine-tuning strategy to improve a nominal controller that was designed in simulation. However, in current practice reinforcement learning approaches remain too data-inefficient, difficult to tune, and prone to robustness issues to be seen as a reliable engineering tools.

The goal of this dissertation is to reconcile these black-box techniques with analytical design principles from control theory, demonstrating how insights gleaned from simplified approximate dynamics models can be used to design data-driven control strategies which are scalable, interpretable and robust by design. In particular, this dissertation draws on tools from geometric control, optimal control and machine learning. Before providing an overview of the dissertation, we will provide a brief historical overview which highlights the strengths and weakness of these approaches.

Geometric Control, Optimal Control and Machine Learining

Geometric control is the culmination of classical 'pen-and-paper' approaches to controller synthesis, and loosely refers to a broad suite of analytical techniques [91, 41]. The roots of these approaches can be traced back to foundational perspectives from mechanics, such as those of Lagrange, Hamilton and d'Alembert, and are the natural evolution of early systems approaches based on frequency domain analysis and later the linear state-space approach of Kalman [51]. These approaches explicitly leverage known structures in the dynamics to hand-design a controller for a specific system and task.

By using hierarchies of structured feedback patterns to simplify the form of closed loop dynamics, these approaches make it possible to plan with simplified dynamics models [103]. These techniques are often inherently robust and the analysis needed to implement these methods means they are often amenable to simple, constructive robustness and safety anal-

ysis. However, the analytic power of geometric control is also the source of its greatest limitations, as the analysis required to write a controller down by hand quickly becomes intractable for high-dimensional systems which are required to perform highly dynamic or intricate tasks.

Optimal control refers to a broad set of synthesis techniques which automatically generate a control strategy by optimizing a user specified cost over the space of system trajectories. This includes methods such as model predictive control [66], dynamic programming [14] and even modern reinforcement learning approaches [36]. These approaches are attractive because, at least in principle, they enable the user to encode complex specifications, such as closed-loop stability or safety with cost functions which are relatively simple to write down.

Optimal control also has a long analytical history that can be traced back to the origins of functional analysis and the pillars of classical mechanics mentioned above. Before the advent of digital computers the synthesis of an optimal control also required detailed 'pen and paper 'calculations which often required understanding the interaction between the structure of the system and the performance objective that the user wrote down. For example, by studying variations on the in the famous Brachistochrone problem, engineers have been able to design counter-intuitive high-performance optimal trajectories for jet fighters [18].

In it's modern form, optimal control is founded on basic principles, such as Bellman's dynamic programming principle [11] and Pontryagins maximum principle [80], which are used as the basis for computational approaches to controller synthesis. In principle these approaches can satisfy the attendant optimality conditions automatically by iteratively updating the control strategy. With the ever-growing availability of computational resources over the last few decades, approaches which can scale with these resources have become the dominant optimal control paradigm for controlling complex high-dimensional robotic systems. As this has happened optimal control has moved further and further away from its analytical roots towards a black-box design philosophy.

This shift in philosophy has come with key advantages and disadvantages. As discussed above, as the robotics community has driven towards controlling more and more complex hardware platforms, it has become intractable to analyze every feature of the closed-loop dynamics by hand. Black-box approaches free engineers from having to design every facet of the closed loop dynamics by hand as, at least in principle, methods such as nonlinear model predictive control and reinforcement learning can automatically generate stabilizing controllers. On the other hand, because we do not have convergence proofs for these approaches, except in the special case of linear dynamics and convex costs [26], tuning the cost functions, initializations and hyper parameters for these approaches to induce the desired behavior remains a time consuming process driven by trial and error, even when working in simulated environments where the dynamics are assumed to be known.

Recently we have also seen a push towards data-driven controller synthesis techniques which can bridge the gap between our simulation models and the real world. This has combined with the trends in optimal control discussed above, naturally culminating in an intense interest focus on data-driven optimal control methods. For example, there have

been numerous studies on data-driven model predictive control [86] techniques over the last decade, which are also sometimes called model-based reinforcement learning methods in the machine learning literature [72]. The set of techniques which have garnered the most attention are neural-network based model-free reinforcement learning methods [60, 36], which effectively learn a controller from sequences of input-output data collected from the plant. The incorporation of machine learning techniques has pushed the black-box nature of nonlinear optimal control approaches to an extreme, completely removing the need to reason about the structure of a dynamics model, but requiring engineers to to tune extra layers of unintuitive hyperparameters associated to the learning components of the stack. Moreover, because these approaches do not leverage known structures in the dynamics, the rate at which they learn from real-world data is significantly slower than approaches like adaptive control, which is built around a structured representation of the system dynamics [91].

Overview of Dissertation

The central thesis underlying this dissertation is that the analytical roots of optimal control need to be reworked to make the theory capable of guiding the design and implementation of scalable computational and data-driven approaches. Through three illustrative vignettes, the dissertation demonstrates how concepts from geometric control can be used to leverage known structures in approximate simplified dynamics models to design optimization and learning problems which are easier to solve and intuitive for engineers to manipulate. In each of these approaches, concepts from control theory are used to gain insights into the *global structure* of optimization-based controller synthesis problems. An emphasis is place on leveraging invariants which remain valid under reasonable forms of dynamics uncertainty, while leaving room of the optimization and learning to overcome the complexity of unmodeled, highly nonlinear dynamics. In short, this dissertation uses optimal control theory as a unifying framework for 'pen and paper' and 'black-box' approaches to controller synthesis, enabling engineers to leverage the strengths of both design philosophies.

The design principles introduced in the dissertation can be summarized with the following interconnected themes:

1. **Embedding Model-based Structures in Reinforcement Learning Objectives**: This leads to learning problems that are interpretable, easier to solve, and have built-in algorithm-agnostic robustness guarantees. This theme is exemplified by the work in Chapters 2, 4 and 5.

2. **Building up Reinforcement Learning Strategies Around Approximate Models:** Rather than learning to control the system from scratch, this approach enables RL algorithms to leverage structural information contained in approximate dynamics models and known feedback architectures. This theme is exemplified by the work in Chapter 3.

3. **Towards a Qualitative and Quantitative Theory for Nonlinear Optimal Control**: This work investigates how the inherent geometry of nonlinear systems leads to fundamental limitations for optimization-based and data-driven control, laying theoretical foundations that guide the use of these methods as engineering tools. This theme is exemplified in Chapters 4 and 5.

The technical content of the thesis in Chapters 2-5 is summarized as follows:

1. **Chaper 2 – Lyapunov Design for Robust and Efficient Robotic Reinforcement Learning:** A key challenge in robotics is reasoning about the behavior a controller will produce over long time horizons. This is because important system properties such as stability and safety are defined over infinite intervals of time. Design techniques from geometric control explicitly leverage known system structures to design controllers with desirable long-term behaviors, but break down in situations where there are significant unmodeled dynamics. In principle, RL can handle unmodeled dynamics by learning directly from real-world data. However, in current practice learning desirable long-horizon behavior requires solving a problem with a large *discount factor*, a hyperparameter that controls how far into the future the algorithm plans and that correlates with the amount of data needed to solve the problem.

   The work in this chapter demonstrates how to leverage approximate dynamics models and design techniques from geometric control to design RL objectives that have desirable long-horizon behaviors baked-in. In particular, the introduced approach uses an approximate *control Lyapunov function* designed using the approximate model to 'supervise' reinforcement learning methods. This approach 1) accelerates the convergence of reinforcement learning methods and provides algorithmic agnostic robustness guarantees and 2) enables the learning to 'correct' the approximate control Lyapunov function designed by the user using real-world data.

2. **Chapter 3 – Reinforcement Learning with Simple Models and Low-Level Feedback Controllers:** Another key challenge in data-driven optimal control is understanding how changes in the control policy affect the user-specified objective over long time horizons. Popular RL methods attempt to learn this from scratch by estimating the *policy gradient*, i.e. the gradient of the objective with respect to the parameters of the control policy. Model-based approaches to calculating the policy gradient involve 1) using real-world data to fit an unstructured dynamics model and 2) taking derivatives through the model and controller. While these approaches are very data-efficient, small inaccuracies in approximate models quickly compound when calculating the policy gradient over longer time horizons. Moreover, the unstructured neural network dynamics models typically used for high-dimensional systems often have poorly behaved derivatives, which leads to erratic policy gradient estimates. In contrast, model-free approaches to estimating the policy gradient bypass these challenges completely but are extremely data-inefficient. Both of these approaches also

suffer from the 'exploding-gradients' problem, which causes long-horizon policy optimization problems to become ill-conditioned. This makes tuning the hyperparameters for these methods extremely difficult and fundamentally limits the rate of convergence these approaches can achieve.

This chapter demonstrates how to produce precise policy gradient estimates over long time horizons using a structured, physics-based approximate dynamics model with known feedback patterns. The first step in the proposed design process is to explicitly embed a family of low-level tracking controllers into the policy class. For example, in the hierarchical design examples considered, a neural network outputs the parameters of a spline which the low-level tracking controller then attempts to track. Next, a policy gradient estimator is introduced which 1) uses the current policy to collect trajectories from the real-world system and 2) takes derivatives through the structured model and overall control architecture (including the low-level controller). This approach has several key benefits. Taking derivatives through a structured model ensures policy gradient estimates do not fluctuate wildly. Moreover, differentiating through the low-level controllers a) automatically corrects for errors in the dynamics model to produce policy gradient estimates with reduced bias and variance, and b) leads to well-conditioned optimization landscapes.

3. **Chapter 4 – Computational Bottlenecks for Global Search Methods:** This chapter investigates how the geometric structure of a control system can make it fundamentally difficult for dynamic programming methods to synthesize a controller that achieves certain objectives. In particular, this chapter studies a two degree of freedom cost design process which captures the main ingredients practitioners consider when designing cost functions for optimal control problems. A key step in the design process is choosing what 'outputs' of the system, i.e. features, appear in the cost function. In geometric control, the choice of outputs can lead to what is called either 'minimum-phase' or 'non-minimum-phase' behavior, depending on how the outputs interact with the inherent geometry of the system. Non-minimum-phase behavior leads to difficulties for classical geometric control techniques and appears in many application domains including flight, driving and bipedal locomotion. It is demonstrated that non-minimum-phase behavior fundamentally limits the space of design decisions available to the user and leads to fundamental limitations for dynamic programming methods and, as a byproduct, model predictive control schemes. In particular, non-minimum-phase behavior is show to lead to computational bottlenecks for nonlinear optimal control which lower bound how difficult it is to obtain a stabilizing controller using these methods.

4. **Chapter 5 – The Stability of Nonlinear Receding Horizon Control with Local Descent Methods:** Model predictive control (MPC) strategies repeatedly solve finite-horizon trajectory optimization problems to choose what inputs to apply to the system. Typical stability results require that a (nearly) globally optimal solution

can be found for each of these optimization problems. However, when the system dynamics are nonlinear these problems are non-convex, and the local search algorithms typically used in practice cannot guarantee a globally optimal solution. In particular, techniques such as gradient descent can only be guaranteed to find (approximate) stationary points for each planning problem. This chapter provides geometric sufficient conditions to ensure that the chosen cost function will lead local search algorithms to stabilize the system. Counterexamples demonstrate that when cost functions are not designed properly, MPC schemes can fail to be stabilizing. Connections are drawn to feedback linearization, which is perhaps the most well-studied concept from the geometric control literature.

Organization: Each of the technical chapters are written in a self-contained fashion, including notational preliminaries and relevant related work. Directions for immediate follow up work to each of the vignettes is discussed in the corresponding chapter, while Chapter 6 provides a retrospective on the dissertation and discusses new directions to be addressed in the future.

Chapter 2

Lypunov Design for Robust and Efficient Robotic Reinforcement Learning

A key challenge in robotics is reasoning about the long-horizon behavior induced by a control policy. This is because important system properties such as stability are inherently long-horizon phenomena. In reinforcement learning (RL), the *discount factor* implicitly controls how far into the future policy optimization algorithms plan when optimizing the objective specified by the user. Standard approaches to designing objective functions for robotic RL, such as penalizing the distance to a reference trajectory, inherently require a large discount factor to learn control policies which stabilize the system [81, 30]. Unfortunately, problems with large discount factors can be extremely difficult to solve, often requiring vast data sets and careful tuning of hyper-parameters [28, 14]. As a number of recent success stories have demonstrated [59, 55, 76, 77, 62, 10], ever-increasing computational resources can be used to solve these problems in simulation and deploy the resulting controllers directly on the real-world system. However, because it is impractical to model every detail of complex hardware platforms, achieving the best performance will require learning from real-world data.

This Chapter introduces a cost-shaping framework which enables users to reliably learn stabilizing control policies with small amounts of real-world data by solving problems with small discount factors. Our approach uses *Control Lyapunov Functions* (CLFs), a standard design tool from the control theory literature [8, 99, 6, 7]. CLFs are 'energy-like' functions for the system which reduce the search for a stabilizing controller to a myopic one-step criterion. In particular, any controller which decreases the energy of the CLF at each instance of time will stabilize the system. Thus, CLFs reduce the long-horizon objective of stabilizing the system to a simple one-step condition. When a CLF is available and the dynamics are known, constructive techniques from the control literature can be used to synthesize a stabilizing controller [99, 6, 7]. However, when there is uncertainty in the dynamics, it is difficult to guarantee that a controller will always decrease the value of the CLF, or that we have even

designed a true CLF for the system.

Our approach is to 1) design an approximate CLF for the real-world system using an approximate dynamics model and 2) modify the 'standard' choice of cost functions mentioned above by adding a term which incentivizes controllers which decrease the approximate CLF over time. This technique effectively uses the approximate CLF as supervision for reinforcement learning, enabling the user to embed known system structures into the learning process while retaining the flexibility of RL to overcome dynamics which are difficult or impractical to model. Indeed, as our analysis demonstrates, when our approach is used reinforcement learning algorithms implicitly learn to 'correct' the approximate CLF provided by the user. When the candidate CLF is close to being a true CLF for the system (in a sense we make precise below), a stabilizing controller can be efficiently learned by solving a problem with a small discount factor. Moreover, the addition of the approximate CLF 'robustifies' the search for a stabilizing controller by ensuring that even highly suboptimal policies will stabilize the system. Finally, in situations where it is too difficult to design a nominal CLF by hand, we demonstrate how one can be learned using a simulation model and the standard style of RL objective discussed above. Specifically, we use the value function learned by the RL algorithm as an approximate CLF for the real-world system. Altogether, beyond accelerating and robustifying RL, our approach also expands the applicability of CLF-based design techniques to situations where is difficult to write down a true CLF for the real-world system.

The primary application of our design technique is to develop data-efficient fine-tuning strategies, wherein a nominal controller developed using a simulation model is refined with small amounts of real-world data. We consider a wide range of analytival CLF design techniques, including CLFs which are designed using a linearized reduced-order model for the system and the hybrid systems approach to designing CLFs for dynamic walking which was introduced in [7]. We also demonstrate how, in cases where it is too difficult to write down a CLF by hand, an approximate CLF can be learned with reinforcement learning in simulation by using the value function associated to a problem with the typical style of cost described above and a large discount factor.

2.1 Background and Problem Setting

Throughout the chapter we will consider deterministic discrete-time systems of the form:

$$x_{k+1} = F(x_k, u_k), \tag{2.1}$$

where $x_k \in \mathcal{X} \subset \mathbb{R}^n$ is the state at time k, $u_k \in \mathcal{U} \subset \mathcal{X}$ is the input applied to the system at that time, and $F \colon \mathcal{X} \times \mathcal{U} \to \mathbb{R}^n$ is the transition function for the system. This general nonlinear model is broad enough to cover many important continuous control tasks for robotics. We will let Π denote the space of all control polices $\pi \colon \mathcal{X} \to \mathcal{U}$ for the system. To ease exposition, for our theoretical analysis we will focus on the case where the goal is

to stabilize the system to a single point, namely the origin. Through our examples we will demonstrate how our cost-shaping technique can be leveraged to achieve more complicated tasks, and in Section 2.4 we outline a path for extending our theoretical results to these settings in future work. Here, we review basic notions of stability that are required for the chapter and introduce control Lyapunov functions.

Asymptotic Stability and Lyapunov Theory

Next, we briefly introduce the elements from stability theory and Lyapunov theory which we use extensively throughout the chapter.

Notation and Terminology

We say that a function $W\colon \mathbb{R}^n \to \mathbb{R}$ is *positive definite* if $W(0) = 0$ and $W(x) > 0$ if $x \neq 0$. Let $\alpha\colon [0, \infty) \to [0, \infty)$ be a continuous function. We say that α is in class $\mathcal{K}$ (denoted $\alpha \in \mathcal{K}$) if $\alpha(0) = 0$ and α is strictly increasing. If in addition we have $\alpha(r) \to \infty$ as $r \to \infty$ when we say that α is in class $\mathcal{K}_\infty$ (denoted $\alpha \in \mathcal{K}_\infty$). Let $\beta\colon [0, \infty) \times [0, \infty)$ be a continuous function. We say that β is in class $\mathcal{KL}$ if for each fixed $t \in [0, \infty)$ the function $\beta(\cdot, t)$ is in class $\mathcal{K}$ and for each fixed $r \in [0, \infty)$ we have $\beta(r, t) \to 0$ as $t \to \infty$.

Basic Stability Results

Definition 1. *We say that the closed loop system $x_{k+1} = F(x_k, \pi(x_k))$ is* asymptotically stable on the set $D \subset \mathbb{R}^n$ *if there exists $\beta \in \mathcal{KL}$ such that for each initial condition $x_0 \in D$ and $k \in \mathbb{N}$ the closed-loop trajectory satisfies:*

$$\|x_k\|_2 \leq \beta(\|x_0\|_2, k). \tag{2.2}$$

Analogously, if the preceding condition holds then we say that π asymptotically stabilizes (2.1)*.*

In words, the definition says that π asymptotically stabilizes (2.1) if all trajectories of the closed-loop system $x_{k+1} = F(x_k, \pi(x_k))$ converge to the origin. Asymptotic stability is a difficult property to verify directly as it requires reasoning about the infinite-horizon behavior of trajectories. Lyapunov functions are a powerful analysis tool which can verify asymptotic stability with a 'one-step' criterion:

Definition 2. *We say that the positive definite function $W\colon \mathbb{R}^n \to \mathbb{R}$ is a Lyapunov function for the closed-loop system $x_{k+1} = F(x_k, \pi(x_k))$ if for each $x \in \mathbb{R}^n$ we have:*

$$W(F(x, \pi(x))) - W(x) < 0. \tag{2.3}$$

Intuitively, the Lyapunov function W can be thought of as an energy-like function for the closed loop system $x_{k+1} = F(x_k, \pi(x_k))$. In this light, the condition (2.3) ensures that the 'energy' of the closed-loop system is decreasing at each point in the state-space. This condition guarantees that the closed-loop system is asymptotically stable [90], and is a simple algebraic condition.

Control Lyapunov Functions

Control Lyapunov Functions [8, 99, 6, 7] extend the notion of a Lyapov function to a controlled, open-loop system:

Definition 3. *We say that a positive definite function $W : \mathbb{R}^n \to \mathbb{R}$ is a* Control Lyapunov Function *(CLF) for* (2.1) *if the following condition holds for each $x \in \mathcal{X} \backslash \{0\}$:*

$$\min_{u \in \mathcal{U}} W(F(x, u)) - W(x) < 0. \tag{2.4}$$

The condition (2.4) ensures that for each $x \in \mathcal{X}$ there exists a choice of input which decreases the 'energy' $W(x)$. Any policy which satisfies the one-step condition $W(F(x, \pi(x))) - W(x) < 0$ can be guaranteed to asymptotically stabilize the system [52]. Note that while control Lyapunov functions are defined formally for the open-loop dynamics (2.1), a Lyapunov function is defined for a particular set of closed-loop dynamics. That is, a control Lyapunov function W for the open-loop dynamics $x_{k+1} = F(x_k, u_k)$ becomes a Lyapunov function for the closed-loop dynamics $x_{k+1} = F(x_k, \pi(x_k))$ which can be used to verify the asymptotic stability of the closed-loop dynamics.

Given a CLF for the system, model-based methods constructively synthesize a controller which satisfies this property using either closed-form equations [99] or by solving an online (convex) optimization problem [29, 7] to satisfy (2.4)[1].

While there is no general procedure for designing CLFs by hand for general nonlinear systems, there do exist constructive procedures for designing CLFs for many important classes of robotic systems, such as manipulator arms [6] and robotic walkers [7] using structural properties of the system. Moreover, in our examples we will investigate how a CLF can be learned from a simulation model and how very coarse CLF candidates can be used to accelerate learning a stabilizing controller.

Stability of Dynamic Programming and Reinforcement Learning

Here we investigate how a common class of cost functions found in the literature can be used to learn stabilizing controllers. In particular, we consider a running cost $\ell : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ of

[1] It is worth noting that the majority of model-based approaches actually formulate CLF-based controller in continuous time, and then sample the resulting controller to produce implementable discrete-time controllers. Here, we have chosen to define CLFs directly in discrete time, as this enables us to be more consistent with standard terminology and notation in the reinforcement learning literature.

the form $\ell(x,u) = Q(x) + R(u)$, where $Q\colon \mathcal{X} \to \mathbb{R}$ is the state cost and $R\colon \mathcal{U} \to \mathbb{R}$ is the input cost. Both Q and R are assumed to be positive definite (in practice, both are usually quadratic).

Given a policy $\pi \in \Pi$, discount factor $\gamma \in [0,1]$, and initial condition $x_0 \in \mathcal{X}$, the associated long-run cost is:

$$V_\gamma^\pi(x_0) = \sum_{k=0}^{\infty} \gamma^k \ell(x_k, \pi(x_k)) \quad \text{s.t. } x_{k+1} = F(x_k, \pi(x_k)), \tag{2.5}$$

where $V_\gamma^\pi\colon \mathcal{X} \to \mathbb{R} \cup \{\infty\}$ is the *value function* associated to π.

Small discount factors incentivize policies which greedily optimize a small number of time-steps into the future, while larger discount factors promote policies which reduce the cost in the long-run. We say that a policy $\pi_\gamma^* \in \Pi$ is *optimal* if it achieves the smallest cost from each $x \in \mathcal{X}$:

$$V_\gamma^{\pi_\gamma^*}(x) = V_\gamma^*(x) := \inf_{\pi \in \Pi} V_\gamma^\pi(x), \qquad \forall x \in \mathcal{X},$$

where $V_\gamma^*\colon \mathcal{X} \to \mathbb{R} \cup \{\infty\}$ is the *optimal value function*. To ease exposition throughout the chapter, we will make the following standing Assumptions which are understood to hold even if they are not referenced.

Assumption 1. *For each $\gamma \in [0,1]$ the optimal value function V_γ^* is continuous and bounded on bounded sets.*

Assumption 2. *For each $\gamma \in [0,1]$ there exists at least one optimal policy associated to the cost* (2.5)*.*

These assumptions are standard in the literature, and enable us to avoid certain technical pathologies which would detract from the main results of the chapter. We refer the reader to [81] for a discussion on conditions which will ensure these assumptions hold. Note that these assumptions imply that the there is at least on control policy which stabilizes the system, as if this were not the case then all control policies would incur an infinite cost from some bounded initial condition.

Under these conditions, is well-known [14] that the optimal value function will satisfy the Bellman equation:

$$V_\gamma^*(x) = \inf_{u \in \mathcal{U}} \left[\gamma V_\gamma^*(F(x,u)) + \ell(x,u)\right], \qquad \forall x \in \mathcal{X}, \tag{2.6}$$

and an optimal policy π_γ^* will satisfy

$$\pi_\gamma^*(x) \in \arg\min_{u \in \mathcal{U}} \left[\gamma V_\gamma^*(F(x,u)) + \ell(x,u)\right], \quad \forall x \in \mathcal{X}.$$

Unfortunately, it is impractical to directly search over Π to find a policy which meets these conditions. This necessitates the use of function approximation schemes (e.g. feed-forward neural networks) to instead represent a subset of policies $\hat{\Pi} \subset \Pi$ to search over. Indeed, modern RL approaches for robotics randomly sample the space of trajectories to optimize problems of the form:

$$\inf_{\pi \in \hat{\Pi}} \mathbb{E}_{x_0 \sim X_0}\big[V_\gamma^\pi(x_0)\big], \tag{2.7}$$

where X_0 is a distribution over initial conditions. While this approach enables these methods to optimize high-dimensional policies, they are data-hungry, can display high-variance and thus frequently return highly sub-optimal policies when data is limited. To better understand the effect that this has on the stability of learned policies, for each $\pi \in \hat{\Pi}$ and $\gamma \in [0, 1]$ define the *optimality gap*:

$$\epsilon_\gamma^\pi(x) = V_\gamma^\pi(x) - V_\gamma^*(x).$$

The temporal difference equation [14] dictates that for each $x \in \mathcal{X}$ the policy satisfies:

$$V_\gamma^\pi(x) = \gamma V_\gamma^\pi(F(x, \pi(x))) + \ell(x, \pi(x)). \tag{2.8}$$

From these equations we can obtain:

$$V_\gamma^\pi(F(x, \pi(x))) - V_\gamma^\pi(x) = \frac{1}{\gamma}\big(- \ell(x, \pi(x)) + (1-\gamma)V_\gamma^\pi(x)\big) \tag{2.9}$$

$$= \frac{1}{\gamma}\big(- \ell(x, \pi(x)) + (1-\gamma)[V_\gamma^*(x) + \epsilon_\gamma^\pi(x)]\big) \tag{2.10}$$

$$\leq \frac{1}{\gamma}\big(- Q(x) + (1-\gamma)[V_\gamma^*(x) + \epsilon_\gamma^\pi(x)]\big), \tag{2.11}$$

where we have first rearranged (2.8), then used $V_\gamma^\pi(x) = V_\gamma^*(x) + \epsilon_\gamma^\pi(x)$, and finally we have used $\ell(x, \pi(x)) \geq Q(x)$. Inequalities of this sort are the building block for proving the stability of suboptimal polices in the dynamic programming literature [30, 81].

By inspecting the cost (2.5) we see that V_γ^π is positive definite (since Q is positive definite). Thus, if the right-hand side of (2.11) is negative for each $x \in \mathcal{X} \setminus \{0\}$, this inequality shows that V_γ^π is a CLF for (2.1), and that π is an asymptotically stabilizing control policy. In other words, V_γ^π is a CLF which is implicitly learned during the training process. Indeed, many RL algorithms directly learn an estimate of the value function, a fact which we later exploit to learn a CLF for in cases where it is too difficult to write one down by hand.

Note that the right hand side of (2.11) will only be negative if

$$V_\gamma^*(x) + \epsilon_\gamma^\pi(x) < \frac{1}{1-\gamma} Q(x). \tag{2.12}$$

Since from (2.5) we know that $V_\gamma^*(x) > Q(x)$ for each $x \in \mathcal{X}$, even the optimal policy (which has no optimality gap) will only be stabilizing if γ is large enough. On the other hand, for a fixed $\gamma \in (0, 1]$, this inequality also quantifies how sub-optimal a policy can be while maintaining stability. To make these observations more quantitative we make the following assumption:

Assumption 3. *For each $\gamma \in [0,1]$ there exists $C_\gamma \geq 1$ such that $V_\gamma^*(x) \leq C_\gamma Q(x)$ for each $x \in \mathcal{X}$.*

Growth conditions of this form are standard in the literature on the stability of approximate dynamic programming [64, 81, 30, 31]. Note that, because the running cost ℓ is non-negative, we have $C_{\gamma'} \leq C_{\gamma''}$ if $\gamma' \leq \gamma''$. In particular, the constant C_1 upper-bounds the ratio between the one-step cost and the optimal undiscounted value function. When C_1 is smaller, the optimal undiscounted policy is more 'contractive' and approximate dynamic programming methods converge more rapidly to an optimal solution [64]. Thus, intuitively the constants $C_\gamma \geq 1$ will be smaller when the system is easier to stabilize. The following result is essentially a specialization of the main result from [31]:

Proposition 1. *Let Assumption 3 hold and let $\gamma \in [0,1]$ and $\pi \in \Pi$ be fixed. Further assume that there exists $\delta > 0$ such that for each $x \in \mathcal{X}$ we have i) $\epsilon_\gamma^\pi(x) \leq \delta Q(x)$ and ii) $C_\gamma + \delta < \frac{1}{1-\gamma}$. Then, π asymptotically stabilizes* (2.1).

Proof. Combining conditions *i)* and *ii)* with equation (2.11) yields:

$$V_\gamma^\pi(F(x,\pi(x))) - V_\gamma^\pi(x) \leq \frac{1}{\gamma}\big(-1 + (1-\gamma)[C_\gamma + \delta]\big)Q(x). \qquad \square$$

Thus the RHS of the preceding equation will be negative-definite if $C_\gamma + \delta < \frac{1}{1-\gamma}$, which demonstrates the desired result.

Remark 1. *(Stability Properties of the Cost Function) In the following section we will derive an analogous result to Proposition 1 for the novel reshaped cost function we propose below. When comparing these results we will primarily focus on the effect of the constants $C_\gamma \geq 1$ (and the equivalent constants for the new setting). The C_γ constants can be used to bound how large of a discount factor is need to stabilize the system. In particular, Proposition 1 implies that the optimal policy will stabilize the system for each γ which satisfies $\gamma > 1 - \frac{1}{C_\gamma}$. The C_γ constants also characterizes how 'robust' the cost function is to suboptimal policies. In particular, for a fixed discount factor, the policy will stabilize the system if $\delta < \frac{1}{1-\gamma} - C_\gamma$. Thus smaller values of the C_γ constants permit more suboptimal policies.*

2.2 Lyapunov Design for Infinite Horizon Reinforcement Learning

Our method uses a positive definite candidate Control Lyapunov Function $W : \mathbb{R}^n \to \mathbb{R}$ for the nonlinear dynamics (2.1), and reshapes (2.5) to our proposed new long horizon cost

$\tilde{V}_\gamma^\pi \colon \mathcal{X} \to \mathbb{R} \cup \{\infty\}$:

$$\tilde{V}_\gamma^\pi(x_0) = \sum_{k=0}^{\infty} \gamma^k \Big([W\big(F(x_k, \pi(x_k))\big) - W(x_k)] + \ell(x_k, \pi(x_k)) \Big) \quad \text{s.t. } x_{k+1} = F(x_k, \pi(x_k)). \tag{2.13}$$

As we shall see below, our method works best when W is in fact a CLF for the system, but still provides benefits when it is only an 'approximate' CLF for the system (in a sense we will make precise later). For each $\gamma \in [0, 1]$ the new optimal value function is given by:

$$\tilde{V}_\gamma^*(x) = \inf_{\pi \in \Pi} \tilde{V}_\gamma^\pi(x). \tag{2.14}$$

The new cost (2.13) includes the amount that W changes at each time step, and thus encourages choices of inputs which decrease W over time. In this case, the Bellman equation [14] dictates:

$$\tilde{V}_\gamma^*(x) = \inf_{u \in \mathcal{U}} \big[\gamma \tilde{V}_\gamma^*(F(x, u)) + \Delta W(x, u) + \ell(x, u)\big], \quad \forall x \in \mathcal{X}, \tag{2.15}$$

where we have adopted the short had

$$\Delta W(x, u) := W(F(x, u)) - W(x) \tag{2.16}$$

. To gain some intuition for the approach let us consider the two extremes where $\gamma = 0$ and $\gamma = 1$. In the case where $\gamma = 1$, by inspection we see that $\tilde{V}_1^* = V_1^* - W$ solves the Bellman equation. Plugging in this solution demonstrates that any optimal policy $\tilde{\pi}_1^*$ must satisfy

$$\tilde{\pi}_1^*(x) \in \arg\min_{u \in \mathcal{U}} [V_1^*(F(x, u)) + \ell(x, u)].$$

This is precisely the optimality condition for the original cost (2.5) when $\gamma = 1$, and thus the set of optimal policies for the two problems coincide. Thus, in this case, by embedding the CLF in the cost we are effectively using W as a warm-start initial guess for the optimal value function.

In the other extreme where $\gamma = 0$, from (2.15) we see that an optimal policy must satisfy

$$\tilde{\pi}_0^*(x) \in \arg\min_{u \in \mathcal{U}} \big[\Delta W(x, u) + \ell(x, u)\big].$$

Thus, in this extreme the optimal policy myopically decreases the value of the CLF, weighted against the cost of the the input that is required to do so. Intuitively, when $\gamma = 0$ our objective function is similar to an imitation learning objective, while when $\gamma = 1$ the objective is more similar to the long horizon objectives typically used in reinforcement learning.

Stabilty Analysis

Thus, when $\gamma = 0$ the optimal policy attempts to greedily decrease the value of the candidate CLF and the one-step cost on the input. As we shall see below, when intermediate discount factors are used, optimal policies may instead decrease the value of W over the course of several steps.

Using the new cost function (2.13), each policy must satisfy the new difference equation:

$$\tilde{V}_\gamma^\pi(x) = \gamma \tilde{V}_\gamma^\pi\big(F(x,\pi(x))\big) + W\big(F(x,\pi(x))\big) - W(x) + \ell(x,\pi(x)). \tag{2.17}$$

In our stability analysis, we will use the following composite function as a candidate CLF for (2.1):

$$\tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi(x) = W(x) + \gamma \tilde{V}_\gamma^\pi(x). \tag{2.18}$$

We provide an interpretation of this curious candidate CLF in Remark 2 below, but first perform an initial analysis similar to the one presented in the previous section. Defining for each $\pi \in \hat{\Pi}$, $\gamma \in [0,1]$ and $x \in \mathcal{X}$ the new optimality gap:

$$\tilde{\epsilon}_\gamma^\pi(x) = \tilde{V}_\gamma^*(x) - \tilde{V}_\gamma^\pi(x), \tag{2.19}$$

and following steps analogous to those taken in (2.9)-(2.11), we can obtain the following:

$$\begin{aligned}
\tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi\big(F(x,\pi(x))\big) - \tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi(x) &= -\ell(x,\pi(x)) + (1-\gamma)\tilde{V}_\gamma^\pi(x) && (2.20)\\
&= -\ell(x,\pi(x)) + (1-\gamma)\big[\tilde{V}_\gamma^*(x) + \tilde{\epsilon}_\gamma^\pi(x)\big] && (2.21)\\
&\leq -Q(x) + (1-\gamma)\big[\tilde{V}_\gamma^*(x) + \tilde{\epsilon}_\gamma^\pi(x)\big]. && (2.22)
\end{aligned}$$

Similar to the analysis in the previous section, we will aim to understand when the right-hand side of (2.22) is negative, as this will characterize when π stabilizes the system. One key difference between the inequalities (2.11) and (2.22) is that, while the original value function V_γ^* is necessarily positive definite, $\tilde{V}_\gamma^*$ can actually take on negative values since the addition of the CLF term allows the new running cost in (2.13) to be negative. As we shall see, this forms the basis for the stability and robustness properties our cost formulation enjoys when W is designed properly. Similar to before, we will make the following standing Assumptions throughout the rest of the chapter:

Assumption 4. *For each $\gamma \in [0,1]$ the optimal value function $\tilde{V}_\gamma^*$ is continuous and bounded on bounded sets.*

Assumption 5. *For each $\gamma \in [0,1]$ there exists at least one optimal policy associated to the cost* (2.13)*.*

Remark 2. *(Learning Corrections to W) When the right hand side of* (2.22) *is negative for each $x \in \mathcal{X} \setminus \{0\}$, inequality* (2.22) *demonstrates that $\tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi$ is in fact a CLF for* (2.1) *and that π stabilizes the system (see Theorem 1). We can think of W as an 'initial guess' for a CLF*

for the system, while $\gamma\tilde{V}_\gamma^\pi$ is a 'correction' to W that is implicitly made by a learned policy π. Roughly speaking, the larger the discount factor, the larger this correction. Thus, the user can trade-off how much the learned policy is able to correct the candidate CLF W against the additional complexity of solving a problem with a higher discount factor, depending on how 'good' they believe the CLF candidate to be.

We first state a general stability result for suboptimal policies associated to the new cost, and then discuss how the choice of W affects the stability of suboptimal control policies:

Assumption 6. *For each $\gamma \in [0,1]$ there exists $\tilde{C}_\gamma \in \mathbb{R}$ such that $\tilde{V}_\gamma^*(x) \leq \tilde{C}_\gamma Q(x)$ for each $x \in \mathcal{X}$.*

Because the reshaped one-step cost $W(F(x,u)) - W(x) + \ell(x,u)$ can take on negative values, so can the $\tilde{C}_\gamma$ constants. Moreover, in this case it is possibe to have $\tilde{C}_{\gamma'} \geq \tilde{C}_{\gamma''}$ when $\gamma' \leq \gamma''$. This is because when larger discount factors are used, the optimal policy can benefit from decreasing W further into the future, even when W is not a true CLF for the system that can be decreased at every time-step.

Before stating our main stability result, which uses the function $\boldsymbol{\mathcal{V}}_\gamma^\pi$ to certify the stabilty of the closed-loop system, we first need to verify that this function is a valid candidate Lyapunov function. Namely, we must ensure that it is positive definite:

Lemma 1. *The composite function $\tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi = W + \gamma\tilde{V}_\gamma^\pi \colon \mathcal{X} \to \mathbb{R} \cup \{\infty\}$ is positive definite.*

Proof. Note that we can re-write the reshaped cost (2.13) as

$$\tilde{V}_\gamma^\pi(x_0) = \sum_{k=0}^{\infty} \gamma^k \Big([W(x_{k+1}) - W(x_k) + \ell(x_k, \pi(x_k))] \Big), \tag{2.23}$$

where $\{x_k\}_{k=0}^\infty$ is the state trajectory generated by the policy π from the initial condition $x_0 \in \mathcal{X}$. By rearranging terms we can rewrite this expression as:

$$\tilde{V}_\gamma^\pi(x_0) = -W(x_0) + (1-\gamma)\sum_{k=0}^{\infty} \gamma^k W(x_{k+1}) + \sum_{k=0}^{\infty} \gamma^k \ell(x_k, \pi(x_k)) > -W(x_0) + Q(x_0) \tag{2.24}$$

where we have used the fact that W and ℓ are both non-negative, and that $\ell(x_0, \pi(x_0)) > Q(x_0)$. Thus, using this expression we see that

$$\tilde{\boldsymbol{\mathcal{V}}}_\gamma^\pi(x_0) = W(x_0) + \gamma\tilde{V}_\gamma^\pi(x_0) > (1-\gamma)W(x_0) + \gamma Q(x_0), \tag{2.25}$$

Since Q and W are assumed to be positive definite functions this demonstrates that $\boldsymbol{\mathcal{V}}_\gamma^\pi$ is in fact positive definite, since a convex combination of positive definite functions is positive definite. The proof is concluded by noting that the choice of γ and π is arbitrary, and thus the conclusion that $\boldsymbol{\mathcal{V}}_\gamma^\pi$ is positive definite holds for all policies and discount factors. □

With this result established, we can now introduce our main result, which is analogous to Theorem 1 but for the reshaped cost:

Theorem 1. *Let Assumption 6 hold and let $\gamma \in [0,1]$ and $\pi \in \hat{\Pi}$ be fixed. Further assume that there exists $\tilde{\delta} > 0$ such that for each $x \in \mathcal{X}$ we have i) $\tilde{\epsilon}_\gamma^\pi(x) \leq \delta Q(x)$ and ii) $\tilde{C}_\gamma + \tilde{\delta} < \frac{1}{1-\gamma}$. Then, π asymptotically stabilizes (2.1).*

Proof. Combining conditions *i*) and *ii*) with equation (2.22) yields:

$$\boldsymbol{\mathcal{V}}_\gamma^\pi(F(x,\pi(x))) - \boldsymbol{\mathcal{V}}_\gamma^\pi(x) \leq \big(-1 + (1-\gamma)[\tilde{C}_\gamma + \tilde{\delta}]\big)Q(x). \qquad \square$$

Thus the RHS of the preceding equation will be negative-definite if $\tilde{C}_\gamma + \tilde{\delta} < \frac{1}{1-\gamma}$, which demonstrates the desired result.

As alluded to in Remark 1, we will primarily focus on comparing how large the constants $C_\gamma \geq 1$ and $\tilde{C}_\gamma \in \mathbb{R}$ are for the two problems, as they control the discount factor required to learn a stabilizing policy and also the 'robustness' of the cost to suboptimal controllers. We provide two characterizations which ensure that $\tilde{C}_\gamma < C_\gamma$. The first condition is taken from the model-predictive control literature [44, 34], where CLFs are used as terminal costs for finite-horizon prediction problems.

Lemma 2. *Suppose that for each $x \in \mathcal{X}$ the following condition holds:*

$$\inf_{u \in U} W(F(x,u)) - W(x) + \ell(x,u) \leq 0. \tag{2.26}$$

Then Assumption 6 is satisfied with constant $\tilde{C}_\gamma \leq 0$.

Proof. Consider a policy $\bar{\pi} \in \Pi$ defined for each $x \in \mathcal{X}$ by:

$$\bar{\pi}(x) \in \arg\inf_{u \in \mathcal{U}} W(F(x,u)) - W(x) + \ell(x,u) \leq 0, \tag{2.27}$$

where the preceding inequality follows directly from the assumptions made in the Lemma. Next, for a given initial condition $x_0 \in \mathcal{X}$ let $\{x_k\}_{k=0}^\infty$ be the state trajectory generated by $\bar{\pi}$. The corresponding reshaped cost is given by

$$\begin{aligned}
\tilde{V}_\gamma^{\bar{\pi}}(x_0) &= \sum_{k=0}^\infty \gamma^k \Big([W\big(F(x_k, \bar{\pi}(x_k))\big) - W(x_k)] + \ell(x_k, \bar{\pi}(x_k)) \Big) && (2.28)\\
&\leq \sum_{k=0}^\infty \gamma^k(0) && (2.29)\\
&\leq 0, && (2.30)
\end{aligned}$$

which demonstrates the desired result, since the initial condition and discount factor were chosen arbitrarily. $\square$

The hypothesis of Lemma 2 implies that *i*) W is a true CLF for the system and *ii*) W dominates the running cost ℓ, in the sense that W can be decreased more rapidly than ℓ accumulates at each time-step. Effectively, this condition implies that it is advantageous for polices to myopically decrease W at each time step. Consequently, when this condition holds optimal policies associated to the reshaped costs (2.13) will stabilize the system for any choice of discount factor. Thus, in this case the user has the freedom to either chose a small discount factor and rapidly learn a controller which decreases the value of the CLF, or choose a large discount factor to learn better long-horizon behavior.

The following definition generalizes this condition to cases where W may not be a true CLF for the system but can be decreased over several time-steps:

Definition 4. *We say that the candidate CLF W $\bar{\gamma}$-dominates the running cost ℓ if for each discount factor $\bar{\gamma} \leq \gamma \leq 1$ and $x \in \mathcal{X}$ we have $\tilde{V}_\gamma^*(x) \leq V_\gamma^*(x)$.*

The condition in (4) effectively provides a way of characterizing how 'close' W is to being a true CLF for the real-world system. In particular, the larger $\bar{\gamma}$ the further into the future RL algorithms must look to see the benefits of decreasing W. Our previous discussion, which showed that $\tilde{V}_1^* = V_1^* - W$, demonstrates that every candidate CLF 1-dominates the cost. Moreover, clearly W can only 0-dominate the original cost if it is a CLF for the system. While this condition is more difficult to verify for intermediate values of $\bar{\gamma}$, it provides qualitative insight into how even approximate CLFs for the system can still make it easier to obtain stabilizing controllers.

Remark 3. *(Robustness of reshaped cost) When the condition of Lemma 2 is satisfied we will have $\tilde{C}_\gamma \leq 0 < C_\gamma$, implying the new cost enjoys the desirable robustness properties discussed above. When W satisfies the 'approximate CLF' condition in Definition (4), it will only enjoy these benefits when the discount factor is large enough. We leave it as a matter for future work to provide quantitative estimates for the $\tilde{C}_\gamma$ constants in these regimes, and to provide sufficient conditions which ensure W $\bar{\gamma}$-dominates the running cost. This will require making additional structural assumptions about the difference between the model used to design the CLF and the real world. We have avoided this issue here, since we have explicitly focused on cases where the form of unmodeled real-world dynamics is unknown. Indeed, our goal has been to provide qualitative insights into how to design the fine-tuning strategies we introduce below.*

2.3 Examples and Practical Implementations

We now demonstrate through an extensive series of examples how the proposed methodology can be used to reduce the sample complexity of obtaining a stabilizing controller. A particular focus is given to fine-tuning strategies, wherein a CLF designed (or learned) using and approximate dynamics model is used with our cost formulation to rapidly improve a nominal controller.

In each of the following experiments we report, the soft actor-critic algorithm (SAC) [36] is used as the learning algorithm to optimize the various reward structures we investigate.

Real-World Finetuning with the A1 Quadruped: We apply our approach to train a neural network controller which augments and improves a nominal model-based controller [24] for a quadruped robot using real-world data. For both settings, we use the locomotion controller presented in [24, Section 3.2] as our nominal baseline controller. This controller uses a linearized rigid-body model to formulate a quadratic-program (QP)-based controller to track a desired body pose of the robot. Specifically, the following QP is solved to obtain the ground reaction forces f for the feet in contact with the ground:

$$\begin{aligned} \min_{f} \ & \|\mathbf{M}f - \tilde{g} - \ddot{q}_d\|_Q + \|f\|_R \\ \text{s.t. } & f_z \geq 0, \\ & -\mu f_z \leq f_x \leq \mu f_z, \\ & -\mu f_z \leq f_y \leq \mu f_z, \end{aligned} \tag{2.31}$$

where $\mathbf{M}$ is the inverse inertia matrix of the rigid body, $\tilde{g} := [0, 0, g, 0, 0, 0]$ denotes the acceleration due to gravity and $\ddot{q}_d \in \mathbb{R}^6$ are the desired pose accelerations of the robot's body. In particular, the desired accelerations are obtained using a PD controller,

$$\ddot{q}_d = -k_p(q - q_d) - k_d(\dot{q} - \dot{q}_d), \tag{2.32}$$

with $q \in \mathbb{R}^6$ denoting the robot's body pose. For each expirement, the action space for the neural network is the desired accelerations provided to the based; that is, the neural network provides correction to the target accelerations to induce the desired behavior on the real-world system. We consider two expiremental set-up: one where the network learns to track a wide range of forward velocities and one where the controller learns to compensate for an unknown load that has been attached to the robot.

Velocity Tracking Experiments

As illustrated by the pink curve in Fig. 2.2 (left), the nominal controller fails to accurately track desired velocities specified by the user. We design a CLF around the desired gait using a linearized reduced-order model for the system. We then collect rollouts of $10s$ on the robot hardware with randomly chosen desired velocity profiles, and solve an RL problem using our cost and a discount factor $\gamma = 0$. Our approach is able to learn a policy which significantly improves the tracking performance of the nominal controller within 5 minutes (30 episodes) of hardware data, as shown in Fig. 2.2 (left). A video of these results can be found in *https://youtu.be/l7kBfitE5n8*. Furthermore, in Fig. 2.2 (right) we benchmark our approach in simulation against an RL agent trained with a 'standard' cost which penalizes the squared error with respect to the desired velocity. As this figure demonstrates, our method is able to rapidly decrease the average tracking error in only around 2 thousand steps from the environment. In contrast, the benchmark approach is only able to reach this level of

Figure 2.1: We learn precise tracking policy on hardware for the `Unitree A1` Quadrupedal robot [85] using less than 20 seconds of data. A video of our experiments can be found here *https://youtu.be/l7kBfitE5n8*

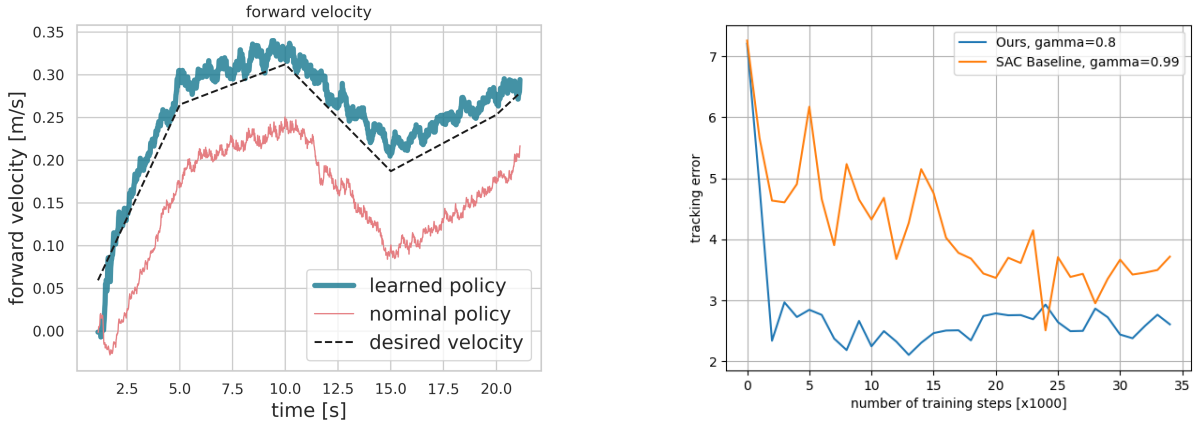

Figure 2.2: (Left) Plot illustrating improved velocity tracking of the learned policy (in dark green) compared to the nominal locomotion controller (in pink) to track a desired velocity profile (in dashed black line) using our proposed method on the `Unitree A1` robot hardware. (Right) Plot from the simulated benchmark study illustrating cumulative velocity tracking error (lower is better) over 10s rollouts at different stages of the training. In orange, we show the results of fine-tuning using SAC with a standard RL cost. In blue, we fine-tune using SAC with our reward reshaping method, with a candidate CLF designed on a nominal linearized model of the robot. In both cases, we plot the results using the discount factor that achieved the best performance.

performance for the first time after around 24 thousand steps, and it takes significantly more data to reach this level of performance consistently.

A1 Quadruped Walking with an Unknown Load:

We attach an un-modeled load to the A1 quadruped, that is equivalent to one-third the mass of the robot. Fine-tuning on hardware the same base controller from the previous set-up where the CLF is designed to stabilize to the target gait, our approach is able to significantly decrease the tracking error to about one-third its nominal value with only one minute of data collected on the robot hardware as illustrated in Fig. 2.3.

To verify that our method out-performs the baseline for this task, we run a simulated

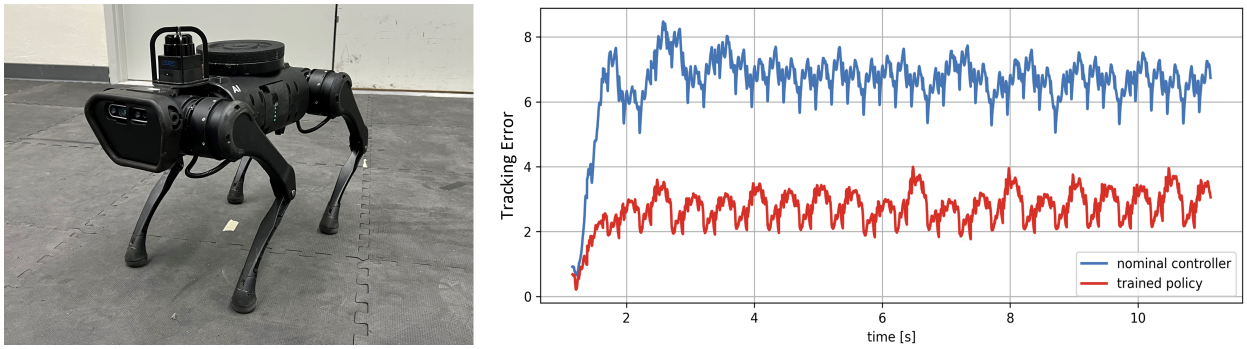

Figure 2.3: Comparison between nominal controller and learned policy after training on 60s of real-world data on the A1 robot with an added 10lb weight. The learned policy is able to significantly reduce the tracking error caused by the added weight.

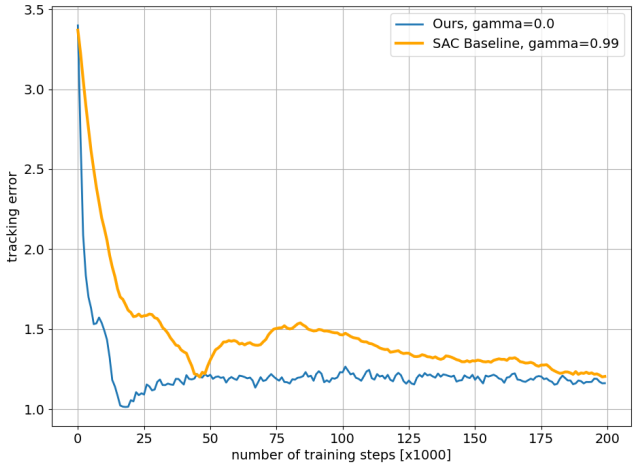

Figure 2.4: Cumulative gait tracking error (lower is better) over 10s rollouts at different stages of the simulated fine-tuning benchmark comparison of the A1 quadruped with an unknown load. In orange, we show the results of fine-tuning using SAC with a standard RL cost which penalizes the distance to the desired gait with a discount factor of $\gamma = 0.99$. In blue, we plot the performance of our cost reshaping method with SAC and a discount factor of $\gamma = 0$. For both cost formulations, we plot the discount factor that led to the best performance.

benchmark comparison similar to the A1 simulation study for velocity tracking that was presented in Section 3.4 of the chapter. For this case, we reproduce the unknown load hardware experiment in simulation by adding a 10lb weight to the robot. When testing our method, we again use SAC with the same reward formulation from the hardware experiments above. For the baseline reward, we penalize the distance to the target that we want to track. Figure 2.4 depicts the best results that we have been able to obtain for each cost formulation across different discount factors and training hyper-parameters. As Fig. 2.4 depicts, our approach quickly converges to a stable walking controller which closely tracks the references after only around 22 thousand steps of the environment. The baseline does not match this performance until it has had access to around 48 thousand steps, and takes much longer to consistently approach the performance of our method.

Fine-tuning a Learned Policy for Cartpole Swing-Up:

We fine-tune a swing-up controller for the `Quanser` cartpole system [84] using real-world data and an initial policy which was pre-trained in simulation but that does not translate well to the real system. Due to the underactuated nature of the system, synthesizing a CLF by hand is challenging. Thus, as alluded to previously, we use a 'typical' cost function of the form (2.5) and a discount factor of $\gamma = 0.999$ to learn a stabilizing neural network policy π_ϕ for a simulation model of the system. As discussed above, we use the value function V_θ associated with the simulation-based policy as the candidate CLF ($W = V_\theta$) for our reward reshaping formulation (2.13). When improving the simulation-based policy π_ϕ with real-world data, we keep the parameters of this network fixed and learn an additional smaller policy π_ψ (so that the overall control action is produced by $\pi_\phi + \pi_\psi$) using our proposed CLF-based cost formulation. We solve the reshaped problem with a discount factor $\gamma = 0$ and collect rollouts of $10s$ on hardware.

Our CLF-based fine-tuning approach is able to successfully complete the swing-up task after collecting data from just one rollout. After collecting data from an additional rollout, the controller is reliable and robust enough to recover from several pushes. A video of these experiments can be found in *https://youtu.be/l7kBfitE5n8*, and plots depicting the states of the system during the successful swing-up motion can be found in 3.1.

Cartpole Fine-tuning Benchmark Comparison

As explained above, previous work has shown that using hardware data to fine-tune a policy that has been pre-trained in simulation is a powerful approach to tackle the sim-2-real gap problem (e.g. [98, 50, 49, 65]). These methods typically take the RL agent trained in simulation and continue its learning process using hardware data, the original cost function and discount factor (see e.g. [98]). In contrast, our proposed approach stops the simulation training of u_ϕ and learns a smaller offset policy u_ψ from hardware data using a separate learning process that has a different reward function $\hat{r}$ (with the CLF candidate being the learned value function in simulation) and a smaller discount factor (in this case $\gamma = 0$).

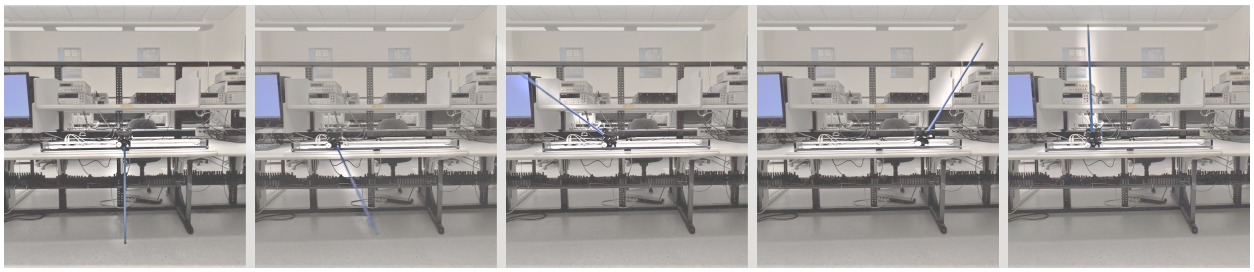

Figure 2.5: We learn precise stabilizing policies on hardware for the `Quanser` cartpole [84] using less than 20 seconds of data. A video of our experiments can be found here *https://youtu.be/l7kBfitE5n8*

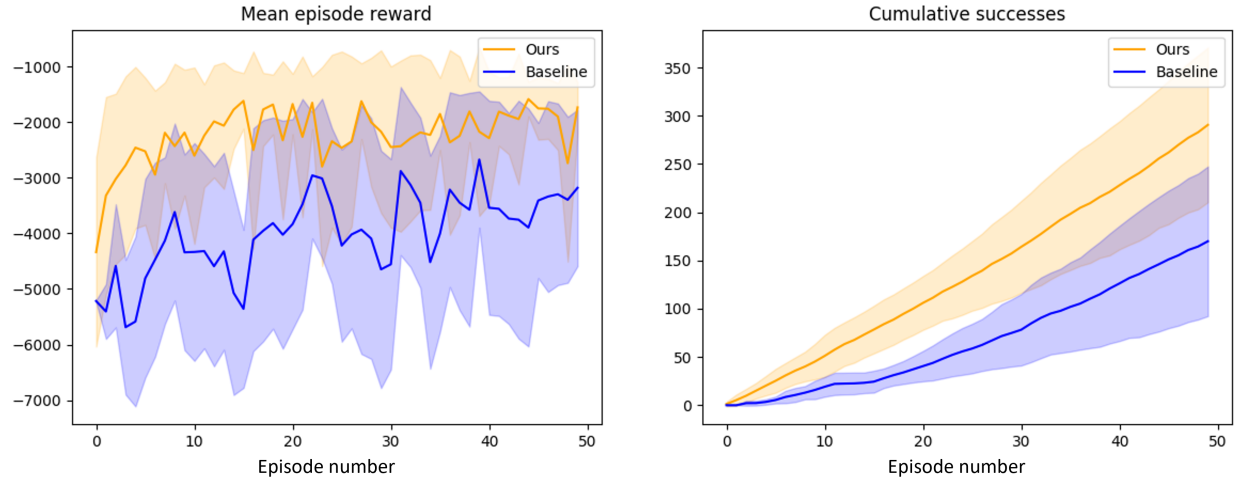

Figure 2.6: Comparison of the simulation results of fine-tuning a cartpole swing-up policy after adding model mismatch. A policy trained on a nominal dynamics model of the cartpole fails when deployed on the new dynamics. In blue, we show the results of continuing to train the agent with the original costs and discount factor. In orange, we fine-tune using our reshaping method with the pre-trained value function and a discount factor of $\gamma = 0$. For each episode of training on the new dynamics model, we compare the performance of both methods when running the cartpole from 10 initial conditions: (on the left) the average original reward without the CLF term, and (on the right) the cumulative number of successful swing-ups. The plots show the mean and standard deviation of the results over 10 different training random seeds.

In Figure 2.6, we compare in simulation the results of using this standard fine-tuning approach with those obtained with our method. For both approaches, we first pre-train a policy π_ϕ and value function V_θ on a nominal set of dynamics using SAC and the reward $r(x_k, u_k) = -0.1\,(5\alpha_k^2 + p_k^2 + 0.05u_k^2) - 5 \cdot 10^3 \cdot \mathbb{1}(|p_k| \geq 0.3)$, and then perturb the parameters of the simulator to introduce model mismatch for the fine-tuning phase. Specifically, we increase the weight and friction of the cart by 200%; and the mass, inertia and length of the pendulum by a 25%. After doing this, we randomly sample 10 initial conditions around the

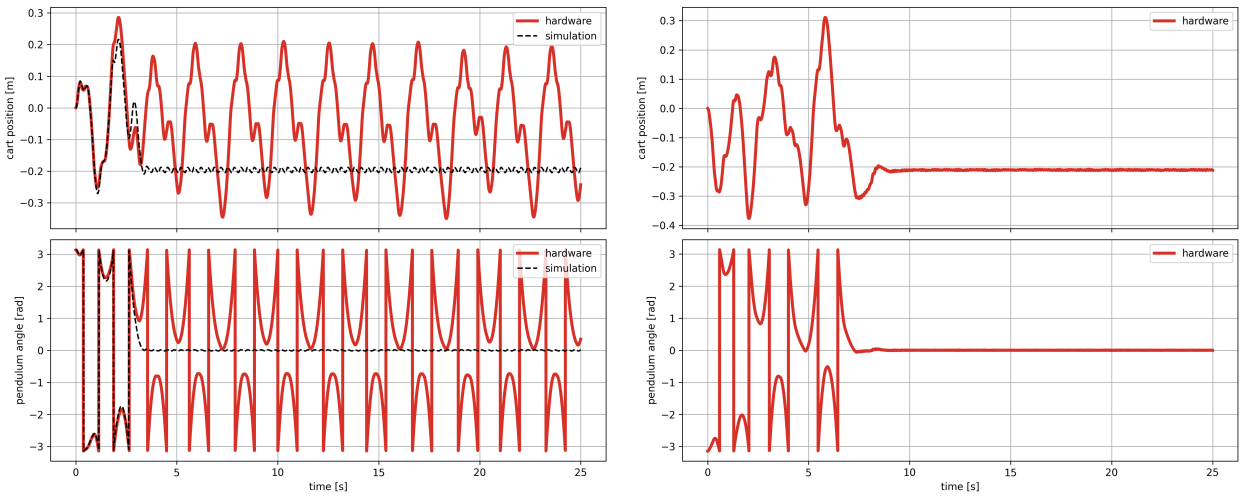

Figure 2.7: Experimental plots of the cart position and pendulum angle of the cartpole system. (left) The policy trained only in simulation fails to bring the real cartpole system to the upright position; (right) by fine-tuning the learned policy with $20s$ of real-world data using our CLF-based reward function, we obtain a successful policy.

downright position ($-0.05m \leq p_0 \leq 0.05m$, $-\pi + 0.05rad \leq \alpha_0 \leq \pi - 0.05rad$, $-0.05m/s \leq \dot{p}_0 \leq 0.05m/s$, $-0.05rad/s \leq \dot{\alpha}_0 \leq 0.05rad/s$). We label a trial as success if within 10 seconds of simulation, the pendulum is stabilized in the set $-0.12rad < \alpha < 0.12rad$, $-0.3rad/s < \dot{\alpha} < 0.3rad/s$ and the cart never gets out of bounds ($|p| < 0.3$). The policy u_ϕ trained with data from the nominal dynamics model does not succeed for any of the 10 initial conditions due to the model mismatch. The baseline in Figure 2.6 is obtained by emptying the replay buffer and using data from the new environment to continue the training process of u_ϕ with the same reward $r(x_k, u_k)$. On the other hand, as with the hardware experiments, our method takes the learned value function V_θ from the nominal dynamics model and learns an offset policy u_ψ using the modified reward $\hat{r}(x_k, u_k) = \Delta V_\theta(x_k, u_k) - 0.1 \cdot (5\alpha_k^2 + p_k^2 + 0.05u_k^2)$. In Figure 2.6, we plot for 10 training random seeds the average original reward $r(x_k, u_k)$ and the cumulative number of successes of the validation episodes ran from the initial conditions mentioned above. The x axis is the number of rollouts of fine-tuning data (each rollout consists of 10 seconds of data). As this figure clearly demonstrates, our approach is able to more rapidly learn a reliable swing-up controller than the baseline. Moreover, as the plot on the left displays, even though we are no longer optimizing for the original reward, by rapidly converging to a stabilizing controller our method still performs better on the original reward than the benchmark.

The above results show that our approach effectively serves to fine-tune policies when the dynamics of the system change. In fact, we have artificially added a severe model mismatch and shown that we can adapt to the new dynamics with a discount factor of 0. This is because

the original value function is still a 'good' CLF candidate for the new system. However, if the change in the dynamics is drastic, or if the overall shape of the motion required to complete the task has to be greatly modified, then the value function from the original dynamics may not be a good CLF candidate, and our method might fail. We have observed that for the cartpole example our method is very robust to variations in the parameters of the cart dynamics (in fact, in the above example we are multiplying both friction and mass of the cart by a factor of 3), but that if we drastically reduce the length and mass of the pendulum by a 50%, our method fails. We hypothesize that this might be related to the underactuated nature of the pendulum dynamics. An interesting direction for future work would therefore be to study under which conditions the original value function retains the CLF properties for a new set of dynamics.

Bipedal Walking Results

Fine-tuning a Bipedal Walking Controller in Simulation: We also apply our design methodology to fine-tune a model-based walking controller [7] for a bipedal robot with large amounts of dynamics uncertainty. Model uncertainty is introduced by doubling the mass of each link of the robot. The nominal controller fails to stabilize the gait and falls within a few steps. To apply our method, we design a CLF around the target gait as in [7] to be used in our reward formulation. As a benchmark comparison, we also train policies with a reward which penalizes the distance to the target motion (no CLF term), as is most commonly done in RL approaches for bipedal locomotion which use target gaits in the reward [62]. Our approach is able to significantly reduce the average tracking error per episode after only 40000 steps of the environment (corresponding to 40 seconds of data), while the baseline does not reach a similar level of performance even after 1.2 million steps, as illustrated in Figure 2.8.

Inverted Pendulum with Input Constraints: Our final example demonstrates the utility of our method even when W is a crude guess for a CLF for the system, through the use of moderate discount factors. We illustrate this for a simple inverted pendulum simulator by varying the magnitude of the input constraints for the system. We use the procedure from [7] to design a candidate CLF for the system. Like many CLF design techniques, this approach assumes there are no input constraints and encourages the pendulum to swing directly up. As the input constraints are tightened, W becomes a poorer candidate CLF, as there is not enough actuation authority to decrease W at each time step. Even in this case, in line with the discussion of Remark 3, if a proper discount factor is used, the addition of the candidate CLF in the reward enables our method to rapidly learn a stabilizing controller for each setting of the input bound.

The states of the system are $x = (\theta, \dot{\theta}) \in \mathbb{R}^2$, where θ is the angle of the arm from the vertical position, and the input $u \in \mathbb{R}$ is the torque applied to the joint. In each of the reinforcement learning experiments reported in Section 3.4 for this system we sample initial conditions over the range $-\pi \leq \theta \leq \pi$ and $-0.1 < \dot{\theta} < 0.1$.

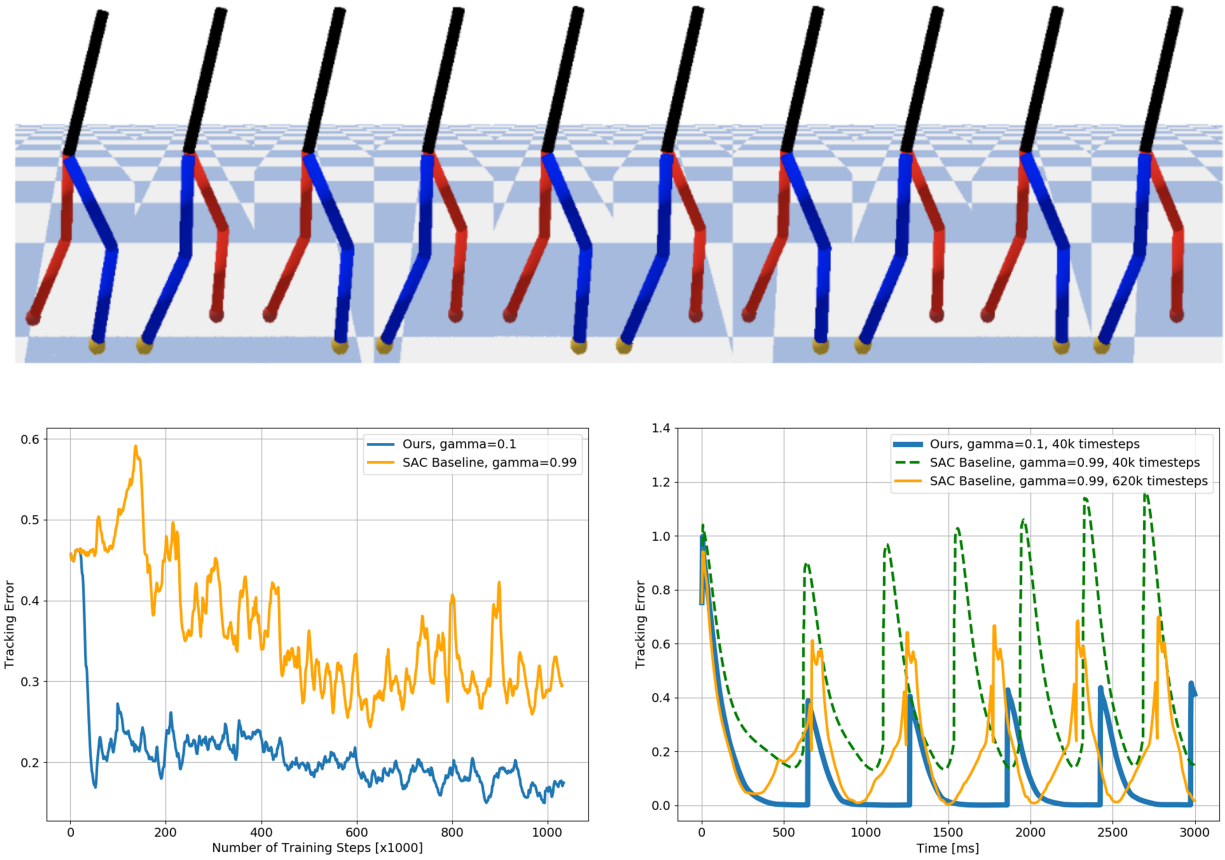

Figure 2.8: (Top) Snapshots of RABBIT [20], a five-link bipedal robot, successfully walking with our learned controller in the PyBullet simulator [23]. (Bottom-Left) Average tracking error (lower is better) per episode at different stages of the training process when fine-tuning a model-based walking controller under model mismatch. In blue, using our CLF-based reward formulation and SAC, the robot learns a stable walking gait with only 40k steps (40 seconds) of training data. In orange, with a baseline that uses a typical reward penalizing the tracking error to the target gait, the training takes longer to converge and does not achieve the same performance. The results show the best performance for both method across different discount factors and training hyper-parameters. (Bottom-Right) Comparison of the tracking error of roll-outs of different learned walking policies. In blue, a policy learned with 40k steps of the environment using our CLF-based reward. In dashed green, a policy learned using the baseline reward with 40k steps of the environment. In orange, a policy learned using the baseline reward with 620k steps of the environment (best baseline policy). The jumps in tracking error occur at the swing-leg impact times. The policy learned with our reward formulation clearly outperforms the baseline, even when the baseline has 15 times as much data.

Training epochs consisted of 5 episodes with 100 simulation steps each, where each time step for the simulator is 0.1 seconds. For both forms of cost function, we sweep across different values of discount factors (from $\gamma = 0$ to $\gamma = 0.95$ in increments of 0.05 and also tried $\gamma = 0.99$) to 1) determine which values of discount factors lead to stabilizing policies and 2) which discount factor allows the agent to learn a stabilizing controller most rapidly. To determine whether a given controller stabilizes the system we randomly sample 20 initial conditions and see if each trajectory reaches the set $\{x \in \mathbb{R}^n : \|x\|_2 < 0.05\}$ within 20 seconds of simulation. For each scenario, the smallest discount factor that lead to a stabilizing controller was also the discount factor that cause the agent to learn a stabilizing controller with the least amount of data.

Training curves for each of the critical values of the discount factor are depicted in Figure 2.9 for each of the cost formulations and input constraints. Each curve indicates the average reward per epoch across 10 different training runs and reports the best results for each scenario after an extensive hyper-parameter sweep. We normalize each training curve so that a reward of 0 indicates the average reward during the first epoch, while a reward of 1 is the largest average reward obtained across all epochs. On each of the training curves the black dot denotes the first training epoch at which a stabilizing controller was obtained.

As illustrated by the plots in Figure 2.9 (a), the addition of the CLF enables our method to more rapidly learn a stabilizing controller in each setting and consistently decreases the amount of data that is needed to learn a stabilizing controller, even when W is not a global CLF for the system. However, the effects are more pronounced when the input constraints are less restrictive and W is a better candidate CLF. For example, when $|u| < 20$ our approach is able to learn a stabilizing controller in 5 iterations, whereas it takes 92 iterations with the original cost (our approach takes $\sim 5.4\%$ as many samples). Meanwhile when $|u| < 4$ our approach takes 198 iterations while the original cost takes 389 iterations (our approach takes $\sim 51\%$ as many samples).Moreover, we observe that larger discount factors are required when $|u| \leq 7$ and $|u| \leq 4$, as W becomes a poorer candidate CLF for these cases.

2.4 Related Work

Discount Factors, Sample Complexity and Reward Shaping: It is well-understood that the discount factor has a significant effect on the size of the data set that RL algorithms need to achieve a desired level of performance. Specifically, it has been shown in numerous contexts [14, 93, 71, 82] that smaller discount factors lead to problems which can be solved more efficiently. This has led to a number of works which explicitly treat the discount factor as a parameter which can be used to control the complexity of the problem alongside reward shaping techniques [48, 78, 28, 100, 19, 74]. Compared to these works, our primary contribution is to demonstrate how CLFs can be combined with model-free algorithms to rapidly learn stabilizing controllers for robotic systems.

Fine-tuning with Real World Data: Recently, there has been much interest in using

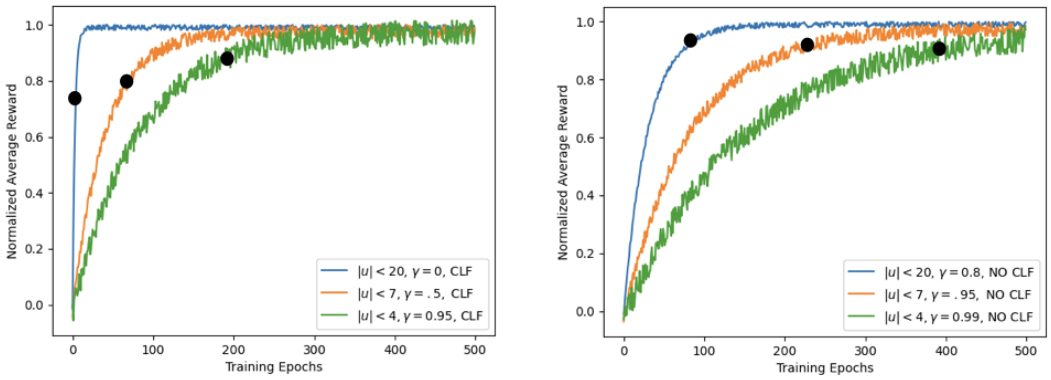

Figure 2.9: Learning curves for an inverted pendulum system under different input constraints. The curves plotted correspond to the smallest discount factors that led to stabilizing policies. On the left, the obtained learning curves use a CLF in the reward. On the right, the reward does not include the CLF term. The black dots denote the first stabilizing policy for each training. For each setting we plot the learning curve for the discount factor that achieved the best performance.

RL to fine-tune policies which have been pre-trained in simulation [98, 50, 49, 65]. These methods typically optimize the same cost function with a large discount factor in both simulation and on the real robot. In contrast, using our cost reshaping techniques, we solve a different problem with a smaller discount factor on hardware which can be solved more efficiently.

Model Predictive Control Literature: Our use of the CLF to reshape the stanard form of cost functions has deep connections to the model predictive control literature (MPC). In their simplest form, MPC control schemes minimize a cost functional of the form

$$\inf_{\hat{\mathbf{u}} \in \mathcal{U}^N} J^N_{MPC}(x_k, \hat{\mathbf{u}}) = \sum_{k=0}^{N-1} \big(Q(\hat{x}_k) + R(\hat{u}_k)\big) + \hat{W}(\hat{x}_N)$$
$$\text{s.t. } \hat{x}_{k+1} = F(\hat{x}_k, \hat{u}_k), \quad \hat{x}_0 = x_k,$$

where x_k is the the current state of the real-world system, $N \in \mathbb{N}$ is the prediction horizon, $\{\hat{x}_k\}_{k=0}^N$ and $\hat{\boldsymbol{u}} = \{\hat{u}_k\}_{k=0}^{N-1} \in \mathcal{U}^N$ are a predictive state trajectory and control sequence, Q and R are as above, and $\hat{W} \colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is the terminal cost which is assumed to be a proper function. The MPC controller then applies the first step of the resulting open loop control and the process repeats, implicitly defining a control law $u_{MPC}(x)$. The MPC cost $J^N_{MPC}(x_k, \cdot)$ can be thought of as a finite-horizon approximation of the original cost (2.5) (except that it is defined over an open-loop sequence of control inputs instead of being a cost over policies).

Stability results from the MPC literature focus primarily on the effects of the prediction horizon N and the choice of terminal cost $\hat{W}$. Under mild conditions, for any choice of

terminal cost (including $\hat{W}(\cdot) \equiv 0$), the user can guarantee that the MPC scheme stabilizes the system on any desired operating region by making the prediction horizon N sufficiently large [42, 34]. Thus, there is a clear connection between the explicit prediction horizon N in MPC schemes and the discount factor γ, as both need to be sufficiently large if a stabilizing controller is to be obtained (since trajectory optimization problems with longer time horizons are generally more difficult to solve). Indeed, in [81] it was pointed out that the *implicit prediction horizon* $\frac{1}{1-\gamma}$, a factor which shows up in the stability conditions in Proposition 1, plays essentially the same role in stability analysis as N for an MPC scheme with no terminal cost when the running cost is $\ell = Q + R$. Thus, much like the 'typical' policy optimization problems discussed in Section 2.1, MPC schemes with no terminal cost (or one which is chosen poorly) may require an excessively long prediction horizon to stabilize the system.

Fortunately, the MPC literature has a well-established technique for reducing the prediction horizon needed to stabilize the system: use an (approximate) CLF for the terminal cost $\hat{W}$ [45, 42, 34]. Indeed, roughly speaking, these results guarantee that for *any prediction horizon* $N \in \mathbb{N}$ the MPC scheme will be stabilizing if $\hat{W}$ is a valid CLF for the system. Extensive empirical evidence [44] and formal analysis [45] has demonstrated that well-designed CLF terminal costs reduce the prediction horizon needed to stabilize the system on a desired set and increase the robustness of the overall MPC control scheme [33]. Thus, in many ways our cost-reshaping approach can be seen as a way to obtain these benefits in the context of infinite horizon model-free reinforcement learning. On the other hand, the design philosophy underlying our approach, in particular our fine-tuning strategy, is quite different. Indeed, the perspective that the learning and optimization is meant to overcome errors in the approximate CLF using real world data represents a sharp departure from the MPC literature.

Future Directions

Other Certificate Functions: In this chapter we have focused on learning stabilizing controllers with CLFs. This is simply a reflection of the fact that control Lyapunov functions are the correct technical object to work with when demonstrating how users can bake in prior information about stabilizing controllers into a cost function. However, the broader theme that 1) model-based design principles can be used to accelerate learning and 2) learning can be used to correct errors in these design pipelines that occur due to unmodeled dynamics provides a broad conceptual framework for future work. Indeed, control Lyapunov functions are just one example of 'certificate' functions which encode desireable long-horizon behaviors with a myopic one-step criterion. For example, previous work of the author and his collaborators investigated how to embed control Barrier functions, which encode safety with respect to state constraint, into costs in an design procedure analogous to the one presented here [101], even though the perspective on modulating the discount factor to correct for model mismatch was not fully developed when this chapter was written. Indeed, there are clear

extension for our approach to any case where a 'certificate' function can be formulated to encode desired behaviors. Moreover, the fine-tuning approach we propose, which is motivate initially by stability theory here, is clearly applicable throughout robotics. Future work will focus on pushing the boundary of this design philosophy as discussed in more detail below.

Convergence Analysis for Approximate Dynamic Programming Methods: In this chapter we have essentially studied the properties of the optimal controllers associated to our reshaped cost functions, in the sense that we characterize how robust this ideal behavior is to varying levels of suboptimality. An interesting avenue for future work is to study whether the CLF term robustifies search algorithms such as approximate policy iteration, approximate value iteration, and approximate Q-learning – which are the prototypical general, abstract approaches that cover most modern reinforcment learning appraoches –in the sense that it corrects for errors made by these algorithms at each step in the training process. Understanding how errors accumulate at each iteration of the algorithm is the standard approach for studying the convergence of these methods [14].

Offline Reinforcement Learning: Offline reinforcement learning, wherein all training for the policy uses a fixed data set which is available *a priori*, has received much interest in recent years [61]. The appeal of this approach is that it opens the possibility for training on vast data sets that are potentially collected from a large number of robots performing a wide range of different tasks. However, learning from such data sets is usually extremely difficult, as a lack of 'on-policy' data can make it difficult to estimate how to improve the current controller. On the other hand, it is well-known that solving problems with large discount factor alleviates these concerns to some extent, as it removes the need for understanding how changes in the control policy will affect the trajectories of the system far into the future [100]. Thus, the fine-tuning approach here has the potential to make it significantly easier to learn from offline data and scale up robotic reinforcement learning to new, unprecedented levels.

Chapter 3

Reinforcement Learning with Simple Dynamics Models and Low-Level Feedback Controllers

Many robotic and autonomous systems display rich dynamics which are impractical to model in full detail using first principles. This has recently led to a resurgence of interest in both 'model-based' and 'model-free' reinforcement learning methods which make use of flexible neural function approximation schemes. These methods are attractive because they can optimize general performance criteria and, at least in principle, eschew the need for designing structured dynamics models and control policies. However, despite achieving impressive performance in a number of application domains [62, 38], the inability of these methods to leverage known system structures leaves them with several key shortcomings.

The most popular model-free reinforcement learning methods [94, 35] for continuous control can be viewed as variations of approximate dynamic programming [14]. These methods internalize the data collected from the real-world system in either a value function or Q-function, which is then used for sampling-based policy gradient methods. These approaches remove the need for an explicit predictive model at the cost of poor sample efficiency and high-variance updates. While these methods achieve good asymptotic performance [73], they require inordinate amounts of real-world data for many applications and may be too unreliable for many safety-critical scenarios.

In contrast, model-based reinforcement learning makes efficient use of the collected data by fitting a predictive model which is used for downstream optimization-based controller synthesis [58, 47, 22]. The key challenge for these approaches is estimating the costs that will be incurred by different sequences of inputs over long time-horizons, as small modeling errors quickly compound and degrade the accuracy of future state estimates [22], especially when the system is open-loop unstable [57]. Errors in the model derivatives are similarly magnified when estimating the sensitivity of state trajectories (and thus the cost function) with respect to changes in the input. Due to these limitations, the most successful model-based

reinforcement learning approaches plan through probabilistic notions of uncertainty to avoid over-exploiting inaccuracies in the learned model [47, 25, 22], but in doing so limit the precision and performance of these methods. Early successes used simple function approximation schemes such as Gaussian processes alongside uncertainty-aware policy derivative algorithms [25]. More recent approaches aimed at high-dimensional control fit neural network models [22, 73]. However, experience has shown that these models often have highly chaotic derivatives [75], making it impractical to employ powerful derivative-based optimization schemes, and leading to the use of conservative sampling-based methods [56, 47].

Both of these paradigms lie in sharp contrast to methods which incorporate structured models into policy updates, such as model-based Iterative Learning Control [5]. For example, 'norm-optimal' or 'optimization-based' iterative learning control [5, 4, 92] and related methods [1, 54] optimize a sequence of open-loop controls to achieve a repetitive task by repeatedly 1) unrolling the current controller on the real system and then 2) using the well-behaved derivatives of an analytical approximate model to suggest local improvements. These methods can achieve extremely precise tracking control even when the nominal model is a coarse approximation [92]. By performing these updates along real-world trajectories, the nominal model is not required to make accurate state predictions far into the future [1]. On the other hand, as we discuss in Section 3.3, by optimizing open-loop controls these methods can still magnify inaccuracies in the model derivatives over long time horizons.

In this chapter, we introduce a data-driven policy optimization approach which is capable of training general function approximators such as neural networks to achieve a wide range of tasks with extremely high precision. Our update scheme is built around not only an approximate analytical model but also a structured class of low-level feedback controllers which are embedded in the policy class. The approach performs approximate stochastic gradient descent on the network parameters by 1) collecting a batch of real-world trajectories from the current policy, then 2) taking derivatives through the approximate model and low-level controller to approximate the closed-loop sensitivity of the state trajectories and 3) using these calculations to approximate the gradient of the overall cost. By incorporating low-level feedback controllers into the updates, our approach enables the user to implicitly leverage known feedback structures in the model. On the theoretical side, we demonstrate that taking derivatives *through* a low-level feedback controller is essential for obtaining accurate gradient estimates over long horizons and producing a well-conditioned optimization landscape. Our examples emphasize hierarchical design principles for building up feedback loops around simple models to achieve complex tasks.

Organization: Section 3.1 briefly reviews the key quantities which affect the convergence rate of approximate gradient descent schemes to motivate our approach. In Section 3.2, we introduce our method for approximating the gradients of the cost function. Section 3.3 demonstrates the benefits of optimizing through low-level feedback. Section 3.4 shows how our approach can be integrated with hierarchical feedback design techniques. Finally, Section 3.5 provides an outlook for future work.

3.1 Background on Approximate Gradient Descent

In this section, we briefly discuss results on the convergence rates of approximate gradient descent schemes for non-convex optimization landscapes. A thorough review of this topic is beyond the scope of this chapter, and we refer the reader to [13, 32] for a complete treatment. Here we will only discuss the basic ingredients for these results as motivation for the bounds we provide in Section 3.3, which are to be viewed as the building blocks for specific convergence results.

Suppose we wish to optimize the following objective:

$$\min_{\theta \in \Theta} f(\theta), \tag{3.1}$$

where $\theta \in \Theta \subset \mathbb{R}^p$ are a set of decision parameters and $f \colon \Theta \to \mathbb{R}$ is a twice-continuously differentiable objective. Consider an approximate gradient descent scheme of the form:

$$\theta_{k+1} = \theta - \alpha_k \hat{g}(\theta_k) \tag{3.2}$$

where $\hat{g}(\theta) \approx \nabla f(\theta)$ is a potentially biased, noisy approximation to the gradient and α_k is the step-size. The following three assumptions are the starting point for standard analyses of approximate gradient descent for non-convex landscapes:

Assumption 7. *There exists a constant $L > 0$ such that for each $\theta \in \Theta$ we have: $\|\nabla^2 f(\theta)\| < L$.*

Assumption 8. *There exists a constant $\Delta \geq 0$ such that for each $\theta \in \Theta$ we have: $\left\|\mathbb{E}\big[\hat{g}(\theta)\big] - \nabla f(\theta)\right\| \leq \Delta$.*

Assumption 9. *There exists a constant $\sigma \geq 0$ such that for each $\theta \in \Theta$ we have: $\mathbb{E}\big[\|\hat{g}(\theta) - \mathbb{E}[\hat{g}(\theta)]\|^2\big] \leq \sigma^2$.*

In the preceding assumptions, expectations are understood to be with respect to any randomness in the gradient estimate. Assumption 8 measures the bias of the gradient estimator while Assumption 9 provides a measure of the variance. Together, Assumptions 8 and 9 characterize how good of an estimator $\hat{g}$ is. On the other hand, Assumption 7 characterizes how quickly the gradient of the underlying objective fluctuates, and limits the convergence rate of gradient descent schemes.

For general non-convex optimization landscapes, convergence results can generally only bound the number of iterations needed to reach an approximate stationary point, namely, a point $\theta \in \Theta$ where $\|\nabla f(\theta)\| \leq \epsilon$ for some desired tolerance $\epsilon > 0$ [13]. Assumption 7 limits the step-sizes $\{\alpha_k\}_{k=0}^{\infty}$ which can be used while maintaining the stability of the method. Roughly speaking, the step-size must be on the order of $\alpha_k \approx \frac{1}{L}$ to ensure the method does not diverge, which limits the rate of convergence. Broadly speaking, as the parameter $L > 0$ increases, the number of steps required to reach an ϵ-stationary point also increases. Meanwhile, the bias parameter $\Delta > 0$ limits how close the descent scheme can get to an

exact stationary point of f (namely, how small we can make the parameter $\epsilon > 0$). Finally, when $\sigma > 0$, a decreasing sequence of step-sizes (where $\alpha_k \to 0$) is required for convergence, further limiting the rate of convergence. Stronger insights can be obtained if the objective is (locally or globally) convex or strongly convex. For example, if the objective is strongly convex, the rate of convergence to the global minimizer is controlled by the condition number of the Hessian [32].

3.2 Problem Formulation

We will represent the real-world system we aim to control with the deterministic dynamics:

$$x_{t+1} = F(x_t, u_t), \tag{3.3}$$

while our simplified model for the system is represented with:

$$x_{t+1} = \hat{F}(x_t, u_t), \tag{3.4}$$

and in both cases we have $x_t \in \mathbb{R}^n$ and $u_t \in U \subset \mathbb{R}^m$. Furthermore, we will assume that both $F\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $\hat{F}\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ are twice continuously differentiable. The exact form of the true dynamics (3.3) is unknown, but we will assume that we can run experiments which can be used to tune the parameters of a policy. Specifically, our goal will be to tune the parameters of a class of time-varying policies of the form $\pi^\theta = \{\pi_t^\theta\}_{t=0}^{T-1}$ subject to losses of the form:

$$\begin{aligned} J_T(\theta; x_0, \psi) = Q_T^\psi(x_T) + \sum_{t=0}^{T-1} \left(Q_t^\psi(x_t) + R_t^\psi(u_t)\right) \\ \text{s.t. } x_{t+1} = F(x_t, u_t) \\ u_t = \pi_t^\theta(x_t; x_0, \psi), \end{aligned} \tag{3.5}$$

where $T \in \mathbb{N}$ is the prediction horizon, $\theta \in \Theta \subset \mathbb{R}^p$ are the parameters of the policy (e.g. the parameters of a neural network), $\psi \in \Psi \subset \mathbb{R}^q$ is a parameter representing an objective we wish to solve, and $Q_t^\psi\colon \mathbb{R}^n \to \mathbb{R}$ and $R_t^\psi\colon U \to \mathbb{R}$ are objective-dependent penalties. We consider initial conditions $x_0 \in D \subset \mathbb{R}^n$, and will refer to each pair $(x_0, \psi) \in D \times \Psi$ as a *task*. Finally, for each task (x_0, ψ), $\pi_t^\theta(\cdot; x_0, \psi)\colon \mathbb{R}^n \to U$ is the feedback control we apply at time t when attempting to achieve this task. In our examples, the space Ψ represents different trajectories that we wish to track.

Ultimately, we will propose a batch update scheme to optimize more general multi-task objectives of the form:

$$\mathcal{J}_T(\theta) = \mathbb{E}_{(x_0,\psi)\sim\mathcal{D}} J_T(\theta; x_0, \psi), \tag{3.6}$$

where $\mathcal{D}$ is a probability distribution over the space of tasks $D \times \Psi$. However, we first demonstrate in Section 3.2 how to calculate $\nabla J_T(\theta; x_0, \psi)$ for a fixed task $(x_0, \psi) \in \mathbb{R}^n \times \Psi$

assuming the dynamics are known. Then in Section 3.2 we demonstrate how to approximate this term using a trajectory collected from the real-world system and the derivatives of the approximate model. These approximations are then used in the batch update in Section 3.2.

To ease notation, define for each $(x_0, \psi, \theta) \in D \times \Psi \times \Theta$:

$$\phi^\theta_{t+1}(x_0, \psi) = F\big(\phi^\theta_t(x_0, \psi), \pi^\theta_t(\phi^\theta_t(x_0, \psi))\big), \quad \phi^\theta_0(x_0, \psi) = x_0,$$

to represent the state trajectory generated by the control policy for the true system (3.3).

The Policy Gradient for a Single Task

For the rest of the section fix $(x_0, \Psi, \theta) \in \mathbb{R}^n \times \Psi \times \Theta$, and let $\{x_t\}_{t=0}^T$ and $\{u_t\}_{t=0}^{T-1}$ with $x_t = \phi^\theta_t(x_0, \psi)$ and $u_t = \pi^\theta_t(x_t; x_0, \psi)$ denote the corresponding sequences of states and controls for the real-world system. We will suppress dependencies on (x_0, ψ) and u_t and x_t when our meaning is clear from context. Along this roll-out let us define:

$$A_t = \frac{\partial}{\partial x} F(x_t, u_t) \qquad B_t = \frac{\partial}{\partial u} F(x_t, u_t)$$

$$K_t = \frac{\partial}{\partial x} \pi^\theta_t(x_t; x_0, \psi) \quad q_t = \frac{d}{dx} Q^\psi_t(x_t) \quad r_t = \frac{d}{du} R^\psi_t(u_t).$$

By adopting the short-hand:

$$\frac{\partial x_t}{\partial \theta} := \frac{\partial}{\partial \theta} \phi^\theta_t(x_0, \psi), \quad \frac{\partial u_t}{\partial \theta} := \frac{\partial}{\partial \theta} \pi^\theta_t(\phi^\theta_t(x_0, \psi); x_0, \psi),$$

and applying the chain rule we can calculate the gradient of the objective for the instance (x_0, ψ) as:

$$\nabla J_T(\theta; x_0, \psi) = q_T \cdot \frac{\partial x_T}{\partial \theta} + \sum_{t=1}^{T-1} q_t \cdot \frac{\partial x_t}{\partial \theta} + r_t \cdot \frac{\partial u_t}{\partial \theta}, \tag{3.7}$$

where for each $t \in \{0, T-1\}$ we have:

$$\frac{\partial u_t}{\partial \theta} = \frac{\partial \pi^\theta_t}{\partial \theta} + \frac{\partial \pi^\theta_t}{\partial x} \cdot \frac{\partial x_t}{\partial \theta} = \frac{\partial \pi^\theta_t}{\partial \theta} + K_t \cdot \frac{\partial x_t}{\partial \theta}.$$

Noting that $\frac{\partial x_0}{\partial \theta} = 0$, because the initial condition is independent of the policy parameters, we can calculate $\{\frac{\partial x_t}{\partial \theta}\}_{t=1}^T$ with with the following recursion:

$$\begin{aligned}
\frac{\partial x_{t+1}}{\partial \theta} &= \frac{\partial}{\partial x} F(x_t, u_t) \cdot \frac{\partial x_t}{\partial \theta} + \frac{\partial}{\partial u} F(x_t, u_t) \cdot \frac{\partial u_t}{\partial \theta} \\
&= A_t \frac{\partial x_t}{\partial \theta} + B_t \frac{\partial u_t}{\partial \theta} \\
&= (A_t + B_t K_t) \frac{\partial x_t}{\partial \theta} + B_t \frac{\partial \pi^\theta_t}{\partial \theta}.
\end{aligned}$$

The term $\frac{\partial x_t}{\partial \theta}$ is the closed-loop sensitivity of the state trajectory with respect to θ, while $\frac{\partial u_t}{\partial \theta}$ is closed-loop sensitivity of the input applied by the policy. Let us successively define:

$$A_t^{cl} := A_t + B_t K_t, \quad \Phi_{t,t'} := \prod_{s=t'+1}^{t-1} A_s^{cl}, \tag{3.8}$$

where the $\Phi_{t,t'}$ are the state transition matrices for the linear time varying dynamics (A_t^{cl}, B_t). Using the solution for this LTV system, we can unroll the recursion for $\frac{\partial x_t}{\partial \theta}$ to obtain:

$$\begin{aligned} \frac{\partial x_t}{\partial \theta} &= \sum_{t'=0}^{t-1} \Phi_{t,t'} B_{t'} \frac{\partial \pi_t^\theta}{\partial \theta} \\ \frac{\partial u_t}{\partial \theta} &= \frac{\partial \pi_t^\theta}{\partial \theta} + K_t \sum_{t'=0}^{t-1} \Phi_{t,t'} B_{t'} \frac{\partial \pi_{t'}^\theta}{\partial \theta}. \end{aligned} \tag{3.9}$$

By plugging these expressions into (3.7) and rearranging the order of summations we obtain the expression:

$$\nabla J_T(\theta; x_0, \psi) = \sum_{t=0}^{T-1} \left(p_{t+1} B_t + r_t\right) \cdot \frac{\partial \pi_t^\theta}{\partial \theta}, \tag{3.10}$$

where the *co-state* $p_t \in \mathbb{R}^{1 \times n}$ is given by:

$$p_t = q_T \cdot \Phi_{T,t} + \sum_{s=t+1}^{T-1} \left(q_s + r_s K_s\right) \cdot \Phi_{s,t}, \tag{3.11}$$

and captures the sensitivity of the closed-loop 'cost-to-go' with respect to changes in the state at time t. Note that $\{p_t\}_{t=0}^T$ can be efficiently computed with the 'backwards pass':

$$p_t = p_{t+1} A_t^{cl} + q_t + r_t K_t, \qquad p_T = q_T,$$

as we detail in the Appendix.

The representation of the gradient in (3.10) and (5.6) is particularly useful, as it neatly separates the gradient calculation into quantities which are known and unknown. Because we designed the policy class $\{\pi^\theta\}_{\theta \in \Theta}$ and the costs Q_t^ψ and R_t^ψ, the values of the K_t, q_t, r_t and $\frac{\partial \pi_t^\theta}{\partial \theta}$ terms are all known, with all of the uncertainty coming from the B_t and $\Phi_{s,t}$ terms (which contain unknown A_t^{cl} terms). As we shall see, the most difficult terms to approximate will be the state transition terms $\Phi_{s,t}$ in p_t, as the repeated compositions used to construct these terms can magnify errors in the individual A_t^{cl} terms.

Approximate Gradients Using the Nominal Model

We now demonstrate how to approximate the policy gradient using the approximate model (5.7) and trajectories collected from the real-world system. Letting $\theta \in \Theta$, $x_0 \in \mathbb{R}^n$ and $\psi \in \Psi$ remain fixed, and retaining the notation from the previous section, let us define the following approximations:

$$\hat{A}_t := \frac{\partial}{\partial x}\hat{F}(x_t, u_t) \qquad \hat{B}_t := \frac{\partial}{\partial u}\hat{F}(x_t, u_t).$$

We approximate the gradient $\nabla J_T(\theta; x_0, \psi)$ by replacing A_t with $\hat{A}_t$ and B_t with $\hat{B}_t$ when building up the gradient calculation in (3.10). In more detail, we define:

$$\hat{A}_t^{cl} := \hat{A}_t + \hat{B}_t K_t, \quad \hat{\Phi}_{t,t'} := \prod_{s=t'+1}^{t-1} \hat{A}_s^{cl},$$

and approximate $\nabla J(\theta; x_0, \psi)$ using the estimate:

$$\hat{g}_T(\theta; x_0, \psi) = \sum_{t=0}^{T-1} \left(\hat{p}_{t+1}\hat{B}_t + r_t\right) \cdot \frac{\partial \pi_t^\theta}{\partial \theta}, \tag{3.12}$$

where the approximate co-state $\hat{p}_t \in \mathbb{R}^{1\times n}$ is given by:

$$\hat{p}_t := q_T \cdot \hat{\Phi}_{T,t} + \sum_{s=t+1}^{T-1} \left(q_s + r_s K_s\right) \cdot \hat{\Phi}_{s,t},$$

and can be calculated with the backwards pass:

$$\hat{p}_t = \hat{p}_{t+1}\hat{A}_t^{cl} + q_t + r_t K_t, \qquad \hat{p}_T = q_T. \tag{3.13}$$

We can construct this gradient approximation by 1) running an experiment or 'forwards pass' on the real system (3.3) with the controller π^θ for the chosen task (x_0, ψ) and 2) using the derivatives of the simplified model (5.7) to approximate the gradient of the cost with the 'backwards pass' in (3.13).

One may note that this gradient calculation is equivalent to approximating the true dynamics (3.3) with a time-varying correction of the form:

$$x_{t+1} = \hat{F}(x_t, u_t) + d_t \tag{3.14}$$

where $d_t := F(x_t, u_t) - \hat{F}(x_t, u_t)$, and calculated the gradient of the cost with respect to these perturbed dynamics. This 'local disturbance identification' perspective comes from the ILC literature [4, 92].

Algorithm 1 Batch Updates for Multi-task Objective

Initialize Time horizon $T \in \mathbb{N}$, number of samples per update $N \in \mathbb{N}$, number of iterations $K \in \mathbb{N}$, step sizes $\{\alpha_k\}_{k=0}^{N-1}$ and initial policy parameters $\theta_1 \in \Theta$
for iterations $k = 1, 2, \ldots, K$ **do**
  Sample N tasks $\{(x_0^i, \psi^i)_{i=1}^N\} \sim \mathcal{D}^N$
  for For task $i = 1, 2, \ldots, N$ **do**
    Unroll $x^i = \{\phi_t^{\theta_k}(x_0^i, \psi^i)\}_{t=0}^T$ on (3.3) with $\pi_t^{\theta_k}$
  Estimate $\hat{g}_T^N(\theta_k)$ using (3.16) and trajectories $\{x^i\}_{i=1}^N$
  Update $\theta_{k+1} = \theta_k + \alpha_k \hat{g}_T^N(\theta)$

Batch Updates for Multiple Tasks

The gradient of the multi-task loss (3.6) is given by:

$$\nabla \mathcal{J}_T(\theta) = \mathbb{E}_{(x_0,\psi)\sim\mathcal{D}}\big[\nabla J_T(\theta; x_0, \psi)\big]. \tag{3.15}$$

To approximate this quantity we make use of the task-specific gradient estimates introduced in the previous section, and given a set of N tasks $\{(x_0^i, \psi^i)\}_{i=1}^N$ drawn i.i.d from $\mathcal{D}$ we use the following biased gradient estimator for $\nabla \mathcal{J}_T(\theta)$:

$$\hat{g}_T^N(\theta) = \frac{1}{N}\sum_{i=1}^N \widehat{g}_T(\theta; x_0^i, \psi^i), \tag{3.16}$$

where $\hat{g}_T(\theta; x_0^i, \psi^i)$ approximates $\nabla J_T(\theta; x_0, \psi)$ as in (3.12). As we summarize in Algorithm 1, we can run a batch of N experiments on the real-world system and use this estimator to perform a noisy version of gradient descent on the overall cost $\mathcal{J}_T(\cdot)$.

3.3 The Benefits of Low-Level Feedback

This Section introduces our main technical results, which illustrate the benefits of explicitly embedding low-level feedback controllers into the policy class $\{\pi^\theta\}_{\theta\in\Theta}$. With an eye towards long-horizon problems, we demonstrate how low-level controllers which stabilize the closed-loop linearizations along the system trajectories allow for more accurate gradient calculations and yield better-conditioned optimization landscapes.

As before, we investigate these issues for a single fixed task and then leverage our findings to characterize the multi-task objective (3.6). In particular, note that:

$$\begin{aligned}\bar{g}_T(\theta) :=& \mathbb{E}_{\left((x_0^i,\psi^i)_{i=1}^N\sim D^N\right)}\big[\frac{1}{N}\sum_{i=1}^N \widehat{g}_T(\theta; x_0^i, \psi^i)\big] \\ =& \mathbb{E}_{(x_0,\psi)\sim D}\big[\hat{g}_T(\theta; x_0, \psi)\big],\end{aligned} \tag{3.17}$$

so that the bias of the estimator is given by:

$$\nabla \mathcal{J}_T(\theta) - \bar{g}_T(\theta) = \mathbb{E}_{(x_0,\psi)\sim\mathcal{D}}\big[\nabla J(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)\big]. \tag{3.18}$$

Moreover, by direct calculation we have:

$$\nabla^2 \mathcal{J}_T(\theta) = \mathbb{E}_{(x_0,\psi)\sim\mathcal{D}}\big[\nabla^2 J_T(\theta; x_0, \psi)\big]. \tag{3.19}$$

Thus, we will be able to directly characterize the expected gradient error $\nabla \mathcal{J}_T(\theta) - \bar{g}_T(\theta)$ and Hessian $\nabla^2 \mathcal{J}_T(\theta)$ by first studying the relevant single-task quantities.

Exponential Blow-up of Gradient Errors

In this section we will investigate how small errors between the model derivatives and the derivatives of the true dynamics can rapidly explode over time and degrade the quality of the estimate $\hat{g}_T(\theta; x_0, \psi) \approx \nabla J_T(\theta; x_0)$ as the length of the prediction horizon $T \in \mathbb{N}$ increases. As in Sections 3.2 and 3.2 above, let $(x_0, \psi) \in D \times \Psi$ again denote our fixed task, retaining all previous notation. We focus on how the quality of the approximation:

$$\hat{\Phi}_{t,t'} \hat{B}_{t'} \approx \Phi_{t,t'} B_{t'},$$

is affected by point-wise errors in the dynamics:

$$\Delta A_t^{cl} := \hat{A}_t^{cl} - A_t^{cl}, \quad \Delta B_t := \hat{B}_t - B_t,$$

when building up the overall gradient estimate (3.12). Recall terms of the form $\Phi_{t,t'} B_{t'}$ are the only sources of uncertainty in the gradient of the true cost (3.10). By defining:

$$\Delta \Phi_{t,t'} = \hat{\Phi}_{t,t'} - \Phi_{t,t'},$$

we can re-write:

$$\begin{aligned} \hat{\Phi}_{t,t'} \hat{B}_{t'} - \Phi_{t,t'} B_{t'} &= \Phi_{t,t'} \hat{B}_{t'} + \Delta \Phi_{t,t'} \hat{B}_{t'} - \Phi_{t,t'} B_{t'} \\ &= \Phi_{t,t'} \Delta B_{t'} + \Delta \Phi_{t,t'} \hat{B}_{t'} \\ &= \Phi_{t,t'} \Delta B_{t'} \\ &\quad + \Big(\sum_{s=t'+1}^{t-1} \Phi_{t,s} \Delta A_s^{cl} \hat{\Phi}_{s-1,t'} \Big) \hat{B}_{t'}, \end{aligned} \tag{3.20}$$

where in the last equality we have used the following steps:

$$\begin{aligned} \Delta \Phi_{t,t'} &= \hat{\Phi}_{t,t'+1} \hat{A}_{t'+1}^{cl} - \Phi_{t,t'+1} A_{t'+1}^{cl} \\ &= \Phi_{t,t'+1} \hat{A}_{t'+1}^{cl} - \Phi_{t,t'+1} A_{t'+1}^{cl} + \Delta \Phi_{t,t'+1} \hat{A}_{t'+1}^{cl} \\ &= \Phi_{t,t'+1} \Delta A_{t'+1}^{cl} + \Delta \Phi_{t,t'+1} \hat{A}_{t'+1}^{cl}, \end{aligned}$$

which can be applied recursively to yield:

$$\begin{aligned} \Delta \Phi_{t,t'} &= \sum_{s=t'}^{t-1} \Phi_{t,s+1} \Delta A_s^{cl} \prod_{l=t'+1}^{s} \hat{A}_l^{cl} \\ &= \sum_{s=t'}^{t-1} \Phi_{t,s+1} \Delta A_s^{cl} \hat{\Phi}_{s+1,t'}. \end{aligned}$$

The last equality in (4.81) provides a clear picture of how inaccuracies in the derivatives of the model are propagated over time. For example, when approximating $\hat{\Phi}_{t,t'}\hat{B}_{t,t'} \approx \Phi_{t,t'}B_{t'}$ the error $\Delta B_{t'}$ is magnified by $\Phi_{t,t'}$, while the error $\Delta A^{cl}_{t'+1}$ is magnified by $\Phi_{t,t'+1}$. The extreme case occurs when the linearizations of the closed-loop dynamics for either the real-world system or the approximate model are unstable. Namely, when there exists $M, \alpha > 0$ such that for each $t > t'$ we have either $\|\Phi_{t,t'}\| > M\alpha^{t-t'}$ or $\|\hat{\Phi}_{t,t'}\| > M\alpha^{t-t'}$. In this case the extent to which the point-wise errors ΔB_t and ΔA^{cl}_t are magnified grows exponentially with the time horizon $T \in \mathbb{N}$. This demonstrates that in general the quality of the approximation $\hat{g}_T(\theta; x_0, \psi) \approx \nabla \hat{J}(\theta; x_0, \psi)$ will degrade exponentially with the length of the prediction horizon, as many systems we wish to control are open-loop unstable. We use the following simple example to illustrate this point:

Running Example: Consider the scalar case with true and modeled dynamics given respectively by:

$$x_{t+1} = ax_t + bu_t \quad \text{and} \quad x_{t+1} = \hat{a}x_t + \hat{b}u_t, \tag{3.21}$$

where $a, \hat{a}, b, \hat{b} > 1$ and $x_t \in \mathbb{R}$ and $u \in U \subset \mathbb{R}$, where U is a convex set. For simplicity, suppose we only want to obtain a policy for a single fixed task $(x_0, \psi) \in D \times \Psi$, and that we use a policy class of the form $\pi^\theta_t(x; x_0, \psi) = \bar{u}_t$ where $\theta = (\bar{u}_t, \ldots, \bar{u}_{T-1})$ and $\bar{u}_t \in U$, and we assume that U is convex. In words, the 'policy' in this case is an open-loop sequence of controls $u_t = \bar{u}_t$. In this case, we have $\Phi_{t,t'} = a^{t-t'-1}$ and $\hat{\Phi}_{t,t'} = \hat{a}^{t-t'-1}$. Comparing this with (4.81) gives a sense for how the modeling errors $\Delta B_t = (\hat{b} - b)$ and $\Delta A^{cl}_t = (\hat{a} - a)$ will be magnified by the unstable dynamics. Note that when we use a 'policy' of this form we are effectively using a very basic iterative learning control approach [4, 1].

Ill-conditioned Optimization Landscapes

We next investigate how the Hessian $\nabla^2 J_T(\theta; x_0, \psi)$ can become ill-conditioned as we increase the time horizon $T \in \mathbb{N}$. Again retaining our previous notation for the fixed instance $(x_0, \psi) \in \mathbb{R}^n \times \Psi$ from above, we demonstrate in [102] that the Hessian is given by the following formula:

$$\begin{aligned}\nabla^2 J_T(\theta; x_0, \psi) = &\big(\frac{\partial x_T}{\partial \theta}\big)^T \cdot \nabla^2 Q_T(x_T) \cdot \frac{\partial x_T}{\partial \theta} \\ &+ \sum_{t=0}^{T-1} \big(\frac{\partial x_t}{\partial \theta}\big)^T \cdot \frac{\partial^2}{\partial x^2} H_t(x_t, p_t, \theta) \cdot \frac{\partial x_t}{\partial \theta} \\ &+ 2\sum_{t=0}^{T-1} \big(\frac{\partial x_t}{\partial \theta}\big)^T \cdot \frac{\partial^2}{\partial x \partial \theta} H_t(x_t, p_{t+1}, \theta) \\ &+ \sum_{t=0}^{T-1} \frac{\partial^2}{\partial \theta^2} H_t(x_t, p_{t+1}, \theta),\end{aligned} \tag{3.22}$$

where for each $t \in [0, \dots, T]$ we define the closed-loop *Hamiltonian* $H_t \colon \mathbb{R}^n \times \mathbb{R}^{1\times n} \times \Theta \to \mathbb{R}$ via:

$$H_t(x, p, \theta) = \left\langle p, F(x, \pi_t^\theta(x)) \right\rangle + Q_t^\psi(x) + R_t^\psi(\pi_t^\theta(x)),$$

suppressing dependence on (x_0, ψ) for readability. While this expression is more involved than the gradient calculation, we highlight the dependence of this expression on the sensitivities of the state trajectory $\{\frac{\partial x_t}{\partial \theta}\}_{t=1}^T$, which were calculated explicitly in (3.9). As we show more formally below, when the linearized dynamics are unstable, and the state trajectories are highly sensitive to changes in the policy, the Hessian will grow exponentially with the horizon. Per the discussion in Section 3.3, this will make the optimization landscape more difficult to optimize over. We illustrate this point with our example:

Running Example: Consider again the simple scalar example in equation (3.21) with the previously described policy class and fixed task (x_0, ψ). Consider a state tracking cost of the form $Q_t^\psi(x_t) = (x_t - x_t^\psi)Q(x_t - x_t^\psi)$ where $Q > 0$ and $\{x_t^\psi\}_{t=0}^{T-1}$ represents the desired trajectory. There is no input cost ($R(\cdot) \equiv 0$) for simplicity. For this example one can calculate: $\frac{\partial^2}{\partial x^2} H_t(x_t, p_{t+1}, \theta) = Q$, $\frac{\partial^2}{\partial x \partial \theta} H_t(x_t, p_{t+1}, \theta) = 0$ and $\frac{\partial^2}{\partial \theta^2} H_t(x_t, p_{t+1}, \theta) = 0$ so that $\nabla^2 J_T(\theta; x_0, \psi) = diag\big(\Delta_1, \Delta_2, \dots, \Delta_{T-1}\big)$ where $\Delta_t = J(\theta; x_0, \psi) = Q\sum_{s=t+1}^T a^{s-t} b$. In this case the loss $J(\theta; x_0, \psi)$ is strongly convex, and the condition number of $\nabla^2 J(\theta; x_0, \psi)$ is $\kappa = \frac{\Delta_0}{\Delta_{T-1}}$, which we observe grows exponentially with the time horizon. Per our discussion in Section 3.1, when T is large the ill-conditioning of the loss landscape will require small step-sizes to maintain stability of gradient descent schemes, limiting the rate of convergence.

The Benefits of Low-level Feedback Control

Before formally characterizing the bias of the multi-task gradient estimator $\hat{g}_T^N(\theta)$ and the Hessians $\nabla^2 \mathcal{J}_T(\theta)$ we return to our running example to illustrate the benefits of constructing the policy class around a low-level feedback controller:

Running Example: Let us again consider the example dynamics in (3.21), the task (x_0, ψ) and the tracking costs Q_t^ψ, R_t^ψ considered above. However, now suppose that we construct our policy class to be of the form $\pi_t^\theta(x_t; x_0, \psi) = k(\bar{x}_{d,t} - x_t)$ where the policy 'parameters' are now $\theta = (\bar{x}_{d,t}, \dots, \bar{x}_{d,t}) \in \mathbb{R}^T$. Namely, in this case the policy parameters specify desired positions at different times which are plugged into a low-level proportional tracking controller which penalizes the 'tracking error' $\theta_t - x_t$. In this case the closed loop dynamics are $x_{t+1} = ax_t + bk(\bar{x}_{d,t} - x_t)$ and we have $A_t^{cl} = a - bk$ and $\hat{A}_t^{cl} = \hat{a} - \hat{b}k$. If the gain $k > 0$ is chosen such that $|a - bk| < 1$ and $|\hat{a} - \hat{b}k| < 1$, then the transition matrices $\hat{\Phi}_{t,t'} = (\hat{A}_t^{cl})^{t-t'-1}$ and $\Phi_{t,t'} = (A_t^{cl})^{t-t'-1}$ will both decay exponentially with the difference $t - t'$. Thus, the point-wise errors will not be magnified exponentially for larger time horizons. Moreover, one can show that in this case the Hessian is given by $\nabla^2 J_T(\theta; x_0, \psi) = diag\big(\bar{\Delta}_1, \bar{\Delta}_2, \dots, \bar{\Delta}_{T-1}\big)$, where $\bar{\Delta}_t = Q\sum_{s=t+1}^T (a - bk)^{s-t} b < \frac{1}{1-(a-bk)} Qb$. In this case we see that the loss $J(\theta; x_0, \psi)$ is again strongly convex, but now the condition number of the Hessian $\kappa = \frac{\bar{\Delta}_0}{\bar{\Delta}_{T-1}} < \frac{1}{1-(a-bk)}$ is of constant order for all $T \in \mathbb{N}$, indicating that the loss

landscape can be more aggressively optimized over by gradient descent methods, per the discussion in Section 3.1.

Remark 4. *Note that it is not essential for the low-level feedback controller to precisely track the specified desired trajectory. We only require it to prevent the sensitivity of the state trajectory (and the approximate sensitivities we obtain from the simple model) from 'blowing up' over longer time horizons. However, because the true dynamics are unknown, we cannot guarantee that $\Phi_{t,t'}$ does not grow exponentially without additional structural assumptions. Nonetheless, in many practical scenarios even coarse low-level controllers can reduce the sensitivity of the state trajectories of the real system. Thus, the primary goal of the theoretical results below is to provide qualitative insight into the benefits of optimizing through a family of 'relatively good' feedback controllers, rather than providing specific correctness or safety guarantees.*

Remark 5. *It should be noted that certain types of iterative learning control approaches do incorporate feedback mechanisms into the update scheme [46] . However, these approaches primarily use the feedback law to provide a good initialization for the feed-forward control and to prevent the learning process from drifting too far from the desired trajectory during training. Namely, these methods typically do not differentiate through a structured class of feedback controllers as we do in our two examples below, where we use a neural network to tune the tracking controller for a wide range of problems.*

We are now ready to formalize the preceding intuition, making use of the following technical Assumptions:

Assumption 10. *There exists a compact subset $\bar{D} \subset \mathbb{R}^n$ such that for each instance $(x_0, \psi) \in D \times \Psi$ and policy parameter $\theta \in \Theta$ we have $\phi_t^\theta(x_0, \psi) \in \bar{D}$ for each $t \in \{0, 1, \ldots, T\}$.*

Assumption 11. *The first and second partial derivatives of Q_t, R_t, π_t^θ, F and $\hat{F}$ are bounded on the set $\bar{D} \times \Psi \times \Theta$.*

The next assumption bounds the point-wise errors between the derivatives of the model (5.7) and the real world system (3.3):

Assumption 12. *Let the set $\bar{D}$ be defined as in Assumption 10. There exist constants $\Delta_A, \Delta_B > 0$ such that for each $x \in \bar{D}$, $\psi \in \Psi$ and $\theta \in \theta$ we have for each $t \in \{0, 1, \ldots, T-1\}$:*

$$\|\frac{\partial}{\partial x} F(x, \pi_t^\theta(x; x_0, \psi)) - \frac{\partial}{\partial x} \hat{F}(x, \pi_t^\theta(x; x_0, \psi))\| < \Delta_A$$
$$\|\frac{\partial}{\partial u} F(x, \pi_t^\theta(x; x_0, \psi)) - \frac{\partial}{\partial u} \hat{F}(x, \pi_t^\theta(x; x_0, \psi))\| < \Delta_B$$

In the statement of the following assumption, for each $(x_0, \psi, \theta) \in D \times \Psi \times \Theta$ we will let $\Phi_{t,t'}^{(\theta, x_0, \psi)}$ and $\hat{\Phi}_{t,t'}^{(\theta, x_0, \psi)}$ respectively denote the state transition matrices generated by linearizing the true and approximate dynamics functions along the corresponding state trajectory of the real-world system.

Assumption 13. *There exists $M > 0$ and $\alpha \geq 0$ such that for each $(x_0, \psi, \theta) \in D \times \Psi \times \Theta$ the following holds for $t > t'$:*

$$\max\{\|\Phi_{t,t'}^{(\theta,x_0,\psi)}\|, \|\hat{\Phi}_{t,t'}^{(\theta,x_0,\psi)}\|\} \leq M\alpha^{t-t'-1}.$$

In the following results we will consider the cases where $\alpha > 1$, $\alpha = 1$ and $\alpha < 1$ which correspond, respectively, to cases where the linearizations of the closed-loop dynamics are unstable, marginally stable, and exponentially stable in variation. The following results rely on supportive Lemmas which characterize the gradient estimators and Hessian of the single task losses, whose poofs can be found below in the Appendix. Our first result characterizes the quality of the gradient estimator (3.16) on different time horizons:

Theorem 2. *Let Assumptions 10-13 hold. Then there exists $C, W > 0$ independent of the parameters $T \in \mathbb{N}$, M, Δ_A, Δ_B and $\alpha \in \mathbb{R}$ such that for each $\theta \in \Theta$ we have:*

$$\|\nabla \mathcal{J}_T(\theta) - \bar{g}_T(\theta)\| \leq \begin{cases} CT^2\alpha^T\Delta & \text{if } \alpha > 1 \\ CT^2\Delta & \text{if } \alpha = 1 \\ CT\Delta & \text{if } \alpha < 1, \end{cases}$$

$$\mathbb{E}\left[\|\hat{g}_T^N(\theta) - \bar{g}_T(\theta)\|^2\right] \leq \begin{cases} \frac{WT^4\alpha^{2T}}{N} & \text{if } \alpha > 1 \\ \frac{WT^4}{N} & \text{if } \alpha = 1 \\ \frac{WT^2}{N} & \text{if } \alpha < 1, \end{cases}$$

where we have denoted $\Delta = \max\{\Delta_A, \Delta_B\}$ and the expectation in the preceding formula is taken with respect to the randomness in the tasks used to generate $\hat{g}_T^N(\theta)$.

Proof. We first bound the bias of the gradient:

$$\begin{aligned} \|\nabla \mathcal{J}_T(\theta) - \bar{g}_T(\theta)\| &= \|\mathbb{E}[\nabla J_T(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)]\| \\ &\leq \mathbb{E}[\|\nabla J_T(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)\|] \\ &\leq \sup \|\nabla J_T(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)\|, \end{aligned}$$

where the preceding expectations are over $(x_0, \psi) \sim D \times \Psi$ and the supremum is taken over $(x_0, \psi) \in D \times \Psi$. The desired bound on the bias directly follows by applying the bound on the task-specific gradient errors from Lemma 4 in the Appendix. Next, to bound the variance estimate note that:

$$\begin{aligned} \mathbb{E}[\|\hat{g}_T^N(\theta) - \bar{g}_T(\theta)\|^2] &= \frac{1}{N^2}\sum_{i=1}^{N} \mathbb{E}[\|\hat{g}_T(\theta; x_0, \psi) - \bar{g}_T(\theta)\|^2] \\ &\leq \frac{1}{N} \sup \|\hat{g}_T(\theta; x_0, \psi) - \bar{g}_T(\theta)\|^2 \\ &\leq \frac{4}{N} \sup \|\hat{g}_T(\theta; x_0, \psi)\|^2, \end{aligned}$$

where the first expectation is over $(x_0^i, \psi^i)_{i=1}^N \sim \mathcal{D}^N$, the second is with respect $(x_0, \psi) \sim \mathcal{D}$ and the supremums are over $(x_0, \psi) \in D \times \Psi$. The desired bound on the variance follows via a direct application of Lemma 3 in the Appendix which provides a uniform upper-bound on the gradient estimates. □

The proofs for Lemmas 3 and 4 can be found in [102]. The proof of Lemma 4 uses (4.81) to track how the point-wise errors in the dynamics derivatives are propagated, and reflects the fact that low-level feedback can squash these errors over long time horizons when $\alpha < 1$, and at least does not magnify them in the marginal case where $\alpha = 1$. The proof of Lemma 3 directly works with the form of the gradient in (3.10). While the method of bounding the variance in Theorem 2 is quite crude, as a more refined analysis would require additional structural assumptions, it captures intuition that we should expect the gradient estimate to be less noisy in cases where the state trajectories are less sensitive to changes in the policy parameters. In particular, when $\alpha > 1$ we may require a very large number of samples at each iteration to avoid unacceptable random fluctuations in the policy parameters.

Next, we provide a bound on the Hessian:

Theorem 3. *Let Assumptions 10,11 and 13 hold. Then there exists a constant $K > 0$ independent of the parameters $T \in \mathbb{N}$, M and $\alpha \geq 0$ such that for each $\theta \in \Theta$ we have:*

$$\|\nabla^2 \mathcal{J}_T(\theta)\|_2 \leq \begin{cases} KT^4\alpha^{3T} & \text{if } \alpha > 1 \\ KT^4 & \text{if } \alpha = 1 \\ KT & \text{if } \alpha < 1. \end{cases}$$

Proof. Similar to before we have:

$$\begin{aligned} \|\nabla^2 \mathcal{J}_T(\theta)\| &\leq \mathbb{E}_{(x_0,\psi)\sim\mathcal{D}}[\|\nabla^2 J_T(\theta; x_0, \psi)\|] \\ &\leq \sup_{(x_0,\psi)\in D} \|\nabla^2 J_T(\theta; x_0, \psi)\|. \end{aligned}$$

The desired bound follows from Lemma 5 in the Appendix, which uniformly bounds the task-specific Hessians. □

The proof of Lemma 5 works directly with the form of the Hessian in (3.22), and can also be found in [102]. The bound in Theorem 3 captures the notion that optimizing *through* a low-level feedback controller which makes the state trajectories less sensitive to changes in the policy parameters will lead to a better-behaved optimization landscape. Per the discussion in Section 3.1 this implies that we can more aggressively optimize the objective in this case by using large step-sizes.

3.4 Design Examples

In these examples, we show how we can build our learning strategy up around low-level control schemes, learning controllers which are able to track desired $x - y$ positions in the

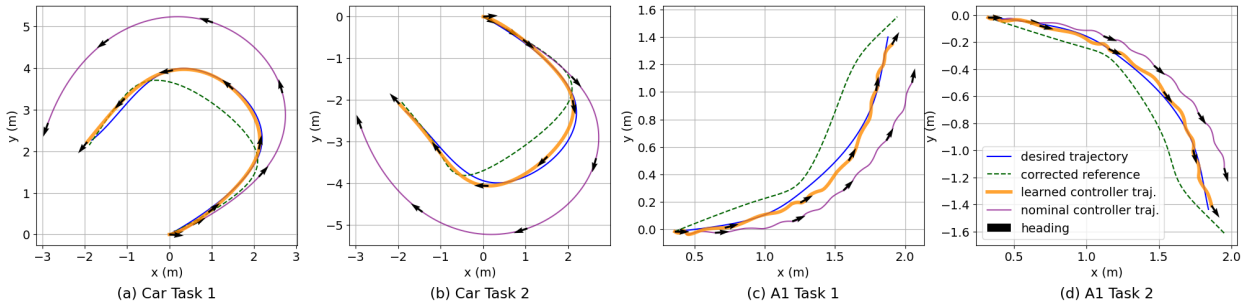

Figure 3.1: Trajectory tracking results for both the car system (a-b) and the A1 quadruped (c-d). For both systems, we show results of 2 different tasks, with reference trajectories plotted in blue. The results show that our method (in orange) using a simple approximate dynamics model learns to track precisely the reference trajectories for both systems, and clearly outperforms the nominal tracking controller (in purple). The modified reference spline from our neural network is plotted in (dashed) green, showing that our method accomplishes the original tracking objective by specifying a deviation on the reference trajectory.

plane for a car and the A1 quadrupedal robot. Even though the low-level dynamics for these systems are extremely different, we are able to apply our approach to essentially the same simple approximate high-level model for both systems.

Derive to Survive – Position Tracking for a Car: We first apply our method to generate accurate tracking controllers for a car. Our analytical dynamics model is:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ v_{t+1} \\ \phi_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + hv_t \cos(\phi_t) \\ y_t + hv_t \sin(\phi_t) \\ v_t + ha_t \\ \phi_t + h\omega_t \end{bmatrix}, \tag{3.23}$$

where $h > 0$ is the discrete time-step, $(x_t, y_t, \phi_t) \in SE(2)$ are the Cartesian coordinates and heading of the car, $v_t \in \mathbb{R}$ is the forward velocity of the car in its local frame, the inputs $(a_t, \omega_t) \in U = [-1, 1] \times [-1, 1]$ and ω_t are respectively the acceleration applied in the forwards direction and the instantaneous turning angle of the wheels. Let $X_t = (x_t, y_t, v_t, \phi_t)$ collect the states for the simple model. Our low-level controller first uses a spline $Z \colon \mathbb{R}^p \to \mathbb{R}^{2\times(T+1)}$ to produce a sequence of desired (x, y)-positions $Z_d = ((x^{\psi}_{d,t}, y^{\psi}_{d,t}))_{t=0}^{T} = Z(\psi) \in \mathbb{R}^{2\times(T+1)}$ for each spline parameter $\psi \in \Psi$ (so that the spline parameters also represent our space of tasks). Then a parameterized family of back-stepping based tracking controllers attempts to track the reference. The tracking controllers are of the form $\{\mu_t\}_{t=0}^{T-1}$ where $\mu_t \colon \mathbb{R}^k \times \mathbb{R}^4 \times \mathbb{R}^{2\times T} \to U$ and produce the control $(a_t, w_t) = \mu_t(G, X_t, Z_d)$, which takes in the current state X_t, the reference trajectory Z_d and a set of feedback gains $G \in \mathbb{R}^k$. To simulate the effects of model uncertainty, we construct the 'true' dynamics by adding drag and errors in how the input enters the dynamics. As shown in Figure 3.1(a-b), the nominal tracking controller does not accurately track the trajectory specified by the spline on the real dynamics. To correct for these deviations, we use a neural network $NN_\theta \colon \mathbb{R}^4 \times \mathbb{R}^p \to \mathbb{R}^k \times \mathbb{R}^p$ with parameters $\theta \in \Theta$ to specify corrections $(\Delta G, \Delta\psi) = (NN^1_\theta(X_0, \psi), NN^2_\theta(X_0, \psi)) = NN_\theta(X_0, \psi)$ to the spline

parameters and feedback gains across a wide range of tasks. Namely, our ultimate policy class $\{\pi^\theta\}_{\theta\in\Theta}$ is of the form $\pi_t^\theta(X_t; X_0, \psi) = \mu_t(\bar{G} + NN_\theta^1(X_0, \psi), X_t, Z(\psi + NN_\theta^2(X_0, \psi)))$ where $\bar{G}$ is the nominal set of feedback gains. Thus, we allow the network to optimize both the feedback gain and the reference to achieve each task. We train the network to track a wide range of dynamic maneuvers, as exemplified in Figure 3.1. The network is a 32×32 MLP with tanh activations, and was trained with 10 rollouts per update over 200 iterations.

Position Tracking for the A1 Quadruped: The A1 quadruped is a 18-DOF robot which must repeatedly make and break contact with the ground to move through the world. Many model-based design approaches for locomotion take a hierarchical approach, building up a multi-loop feedback controller which uses models with varying degrees of simplicity at different layers of abstraction. At the lowest level of abstraction, a Jacobian-based controller converts desired forces at the feet into motor torques. The next layer of abstraction is composed of a 'swing-leg' controller and 'base' controller. This level of abstraction takes in a gait primitive, which specifies a desired sequence of foot-placements and also desired accelerations for the base of the robot. The swing-foot controller specifies desired positions for the feet not currently on the ground using a simple heuristic which is designed to prevent the robot from falling. The base controller takes in desired accelerations for the base and solves a quadratic program to convert this command into forces for the feet currently on the ground. Altogether, these intertwined control loops enable the user to control the robot from a high-level by specifying desired accelerations for the base of the robot. We interface with this controller by using a back-stepping based tracking controller similar to the one proposed for the car to specify desired forward and turning accelerations for the robot base. The model used by the high-level tracking controller is:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ v_{t+1} \\ \phi_{t+1} \\ \omega_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + hv_t\cos(\phi_t) \\ y_t + hv_t\sin(\phi_t) \\ v_t + ha_t^v \\ \phi_t + h\omega_t \\ \omega_t + ha_t^\omega \end{bmatrix}, \tag{3.24}$$

where $(x_t, y_t, \phi_t) \in SE(2)$ are again the Cartesian positions and yaw orientation of the robot in the ground plane, $v_t \in \mathbb{R}$ is again the forward velocity, $\omega_t \in \mathbb{R}$ the turning velocity, and $(a_t^v, a_t^w) \in U \subset \mathbb{R}^2$ are the desired accelerations for the two velocities. Given the desired accelerations from the high-level model, the base controller is then asked to implement these desired accelerations, while using a PD controller to keep the other configuration variables of the base neglected in the high-level model (body height, roll and pitch) at pre-specified desired values. We control the high-level simple model in almost exactly the same way as the car. The only difference is in how the backstepping tracking controller is constructed due to the extra integrator in the yaw coordinate of the new model, but the learned corrections to the reference spline and feedback gains are applied in the same fashion. Figure 3.1(c-d) show that using our proposed method with this simple high-level dynamics model, the A1 robot is able to learn precise tracking policies for across a wide range of desired trajectories, despite the simplicity of the high-level model and the complexity of the underlying dynamics.

Indeed, note the slight periodic oscillations in the trajectories generated by the learned and nominal controller. These come from the complex, periodic contact events that occur in the PyBullet [23] simulator we use for the full 'real-world' dynamics, but do not show up at all in our simple dynamics model. For training we again used a 32×32 MLP network with tanh activations, running 20 rollouts per update for 200 iterations.

3.5 Future Work

There are a number of important directions for future work. The approach presented here is an on-policy approach, which does not make use of data from past iterations when performing updates. Thus, there are many exciting avenues for building more efficient algorithms around our gradient estimation approach. It will also be important to further investigate the limits of our approach by gaining qualitative understanding of when the approximate model and feedback controller are too inaccurate for our method to learn precise controllers. Finally, we aim to apply the principles developed here to more complex models and feedback control loops.

3.6 Additional Proofs

This appendix contains proofs of claims that were omitted in the main document and several supportive Lemmas which were used in the proofs of the main results. Section 3.6 formally derives the costate representation of the gradient us the method of multipliers, while Section 3.6 builds on this calculation to derive the desired representation for the hessian. Finally, Section 3.6 contains the auxiliary Lemmas.

Gradient Calculations

As before, let $\{x_t\}_{t=0}^{T}$ and $\{u_t\}_{t=0}^{T-1}$ denote the state trajectory that results from applying the policy π_θ when applied to the task (x_0, ψ). We will again omit certain dependencies on (x_0, ψ) throughout the section to simplify notation, especially when referring to the policy.

Permitting a slight abuse of notation, we can re-write the cost by moving the dynamics constraints into the cost and weighting them with Lagrange multipliers:

$$J(\theta; x_0, \psi) = Q_T^\psi(x_T) + \sum_{i=0}^{T-1} Q_t^\psi(x_t) + R_t(\pi_\theta^t(x_t)) + p_{t+1}^T\big(x_{t+1} - F(x_t, \pi_\theta^t(x_t))\big) \tag{3.25}$$

Define the Hamiltonian

$$H_t^\psi(x_t, p_{t+1}, \theta) = p_{t+1}^T F(x_t, \pi_\theta^t(x_t)) + Q_t^\psi(x_t) + R_t^\psi(\pi_\theta^t(x_t)), \tag{3.26}$$

and note that we may then re-write the cost as:

$$J(\theta; x_0, \psi) = Q_T^{\psi}(x_T) + \langle p_T, x_T \rangle + \langle p_0, x_0 \rangle + \sum_{t=0}^{T-1} p_t^T x_t - H_t^{\psi}(x_t, p_{t+1}, \theta) \tag{3.27}$$

To reduce clutter below we will frequently omit the arguments from H_t^{ψ}, since it is clear that the map is evaluated at (x_t, p_{t+1}, θ). Let $\delta\theta \in \mathbb{R}^p$ be a variation on the policy parameters and let $\delta x_t = \frac{\partial \phi_\theta^t}{\partial \theta}\delta\theta$ denote the corresponding first variation of the state. To first order, the change in the cost corresponding to these variations is:

$$\delta J|_\theta(\delta\theta) = \langle \nabla Q_T(x_T) + p_T, \delta x_T \rangle + \sum_{t=0}^{T-1} \langle p_t - \nabla_x H_t, \delta x_t \rangle - \langle \nabla_\theta H_t, \delta\theta \rangle. \tag{3.28}$$

To simplify the expression, let us make the following choices for the multipliers:

$$p_T = \nabla Q_T(x_T) \tag{3.29}$$

$$\begin{aligned} p_t^T &= \nabla_x H_t(x_t, p_{t+1}, \theta) && (3.30)\\ &= p_{t+1}^T \frac{\partial}{\partial x} F(x, \pi_\theta^t(x)) + \nabla Q^{\psi}(x_t) + \nabla R^{\psi}(\pi_\theta^T(x_t)) \frac{\partial \pi_\theta^t}{\partial x} && (3.31)\\ &= p_{t+1}^T (A_t + B_t K_t) + q_t + r_t K_t, && (3.32) \end{aligned}$$

where we have applied the short-hand from developed in Section 3.2 for the particular task. Plugging this choice for the multipliers into (3.28) causes the δx_t terms to vanish and yields:

$$\begin{aligned} \delta J|_\theta(\delta\theta) &= \sum_{t=0}^{t-1} \langle \nabla_\theta H_t, \delta\theta \rangle && (3.33)\\ &= \langle p_{t+1}^T \frac{\partial}{\partial u} F(x, \pi_\theta^t) \frac{\partial \pi_\theta^t}{\partial \theta} + \nabla R_t(\pi_\theta^T) \frac{\partial \pi_\theta^t}{\partial \theta}, \delta\theta \rangle && (3.34)\\ &= \sum_{t=0}^{t-1} \langle p_{t+1}^T B_t + r_t, \frac{\partial \pi_\theta^t}{\partial \theta}\delta\theta \rangle && (3.35) \end{aligned}$$

Since this calculation holds for arbitrary $\delta\theta$ this demonstrates that the gradient of the objective is given by:

$$\nabla_\theta J(\theta, x_0) = \sum_{t=0}^{t-1} \langle p_{t+1}^T B_t + r_t, \frac{\partial \pi_\theta^t}{\partial \theta} \rangle. \tag{3.36}$$

Calculating the Hessian

To calculate the Hessian of the objective be continue the Lagrange multiplier approach discussed above. Now let $\delta^2 x_t$ denote the second order variation in the state with respect to

the perturbation $\delta\theta$. By collecting second order terms in (3.27) the the attendant second-order variation to the cost is given by:

$$\delta^2 J|_\theta(\delta\theta) = \langle \delta x_t^T \nabla^2 Q_T^\psi(x_T), \delta x_t\rangle + \langle \nabla Q_T^\psi(x_T) + p_T, \delta^2 x_T\rangle \tag{3.37}$$

$$+ \sum_{t=0}^{T-1} \Big(\langle p_t - \nabla_x H_t^\psi, \delta^2 x_t\rangle + \langle \delta x_t^T \nabla_{xx} H_t^\psi(x_t), \delta x_t\rangle$$

$$+ 2\langle \delta x_t \nabla_{x\theta} H_t^\psi, \delta\theta\rangle + \langle \delta\theta^T \nabla_{\theta\theta} H_t^\psi, \delta\theta\rangle \Big) \tag{3.38}$$

By using the choice of costate introduced above, this time the second order state variations $\delta^2 x_t$ vanish from this expression so that we arrive at:

$$\delta^2 J|_\theta(\delta\theta) = \langle \delta x_t^T \nabla^2 Q_T^\psi(x_T), \delta x_t\rangle \tag{3.39}$$

$$+ \sum_{t=0}^{T-1} \langle \delta x_t^T \nabla_{xx} H_t^\psi(x_t), \delta x_t\rangle + 2\langle \delta x_t \nabla_{x\theta} H_t^\psi, \delta\theta\rangle + \langle \delta\theta^T \nabla_{\theta\theta} H_t^\psi, \delta\theta\rangle,$$

where we recall that we have

$$\delta x_t = \frac{\partial \phi_\theta^t}{\partial \theta} := \sum_{t'=0}^{t-1} \phi_{t,t'} B_{t'} \frac{\partial \pi_\theta^{t'}}{\partial \theta}. \tag{3.40}$$

Supportive Lemmas

Lemma 3. *Let Assumptions 10-13 hold. Then there exists $\beta > 0$ independent of the parameters $T \in \mathbb{N}$, M and $\alpha \in \mathbb{R}$ such that for each $x_0 \in D$, $\psi \in \Psi$ and $\theta \in \Theta$ we have:*

$$\|\nabla_\theta J_T(\theta; x_0, \psi)\| \leq \begin{cases} \beta T^2 \alpha^T & \text{if } \alpha > 1 \\ \beta T^2 & \text{if } \alpha = 1 \\ \beta T & \text{if } \alpha < 1. \end{cases}$$

Proof. Let the constant $L > 0$ be large enough so that it upper-bounds the norm of the first and second partial derivatives of Q_t, R_t, π_t^θ, F and $\hat{F}$ on the set $\bar{D} \times \Psi \times \Theta$. Note that the existence of such a constant is stipulated by Assumption 11. Fix a specific task (x_0, ψ) and set of policy parameters θ and let A_t, B_t, K_t, q_t and r_t be defined along the corresponding trajectory, as in Section 3.2. Recall from Section 3.2 that

$$\nabla J_T(\theta; x_0, \psi) = \sum_{t=0}^{T-1} \big(p_{t+1} B_t + r_t\big) \cdot \frac{\partial \pi_t^\theta}{\partial \theta},$$

where the *co-state* $p_t \in \mathbb{R}^{1\times n}$ is given by:

$$p_t = q_T \cdot \Phi_{T,t} + \sum_{s=t+1}^{T-1} \big(q_s + r_s K_s\big) \cdot \Phi_{s,t}.$$

Thus, we may upper-bound the growth of the co-state as follows:

$$\|p_t\| \leq LM\alpha^{T-t} + \sum_{s=t+1}^{T-1} (L + L^2)M\alpha^{s-t} \tag{3.41}$$

By carrying out the summation, we observe that there exists $C_1 > 0$ sufficiently large such that

$$\|p_t\| \leq \begin{cases} C_1 T\alpha^T & \text{if } \alpha > 1 \\ C_1 T & \text{if } \alpha = 1 \\ C_1 & \text{if } \alpha < 1, \end{cases} \tag{3.42}$$

where we have used the fact that $\sum_{s=t+1}^{T-1} M\alpha^{s-t} < M\frac{1}{1-\alpha}$ for the third case. We can bound the overall gradient as follows:

$$\|\nabla J_T(\theta; x_0, \psi)\| = \sum_{t=0}^{T-1} L\big(L\|p_{t+1}\| + L\big), \tag{3.43}$$

which when combined with the bound on the costate above demonstrates the desired result for some constant $\beta > 0$ sufficiently large to cover all choices of (x_0, ψ).

□

Lemma 4. *Let Assumptions 10-13 hold. Then there exists $C > 0$ independent of $T \in \mathbb{N}$, $M, \Delta_A, \Delta_B > 0$ and $\alpha \in \mathbb{R}$ such that for each $x_0 \in D$, $\psi \in \Psi$ and $\theta \in \Theta$ we have:*

$$\|\nabla_\theta J_T(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)\| \leq \begin{cases} CT^3\alpha^T\Delta & \text{if } \alpha > 1 \\ CT^3\Delta & \text{if } \alpha = 1 \\ CT^2\Delta & \text{if } \alpha < 1, \end{cases}$$

where $\Delta = \min\{\Delta_A, \Delta_B\}$.

Proof. Let the constant $L > 0$ be large enough so that it upper-bounds the norm of the first and second partial derivatives of Q_t, R_t, π_t^θ, F and $\hat{F}$ on the set $\bar{D} \times \Psi \times \Theta$. Note that the existence of such a constant is stipulated by Assumption 11. Fix a specific task (x_0, ψ) and set of policy parameters θ and let A_t, B_t, K_t, q_t and r_t be defined along the corresponding trajectory, as in Section 3.2.

Using equations (3.12), (3.10) and (4.81) we obtain:

$$\begin{aligned} \|\nabla J_T(\theta; x_0, \psi) - \hat{g}_T(\theta, x_0, \psi)\| &= \|\sum_{t=1}^{T} q_t \cdot \sum_{t'=0}^{t} (\Phi_{t,t'}B_{t'} - \hat{\Phi}_{t,t'}\hat{B}_{t'})\| \\ &\leq \sum_{t=1}^{T} \|q_t\| \cdot \sum_{t'=0}^{t} \|\Phi_{t,t'}\Delta B_{t'} + \big(\sum_{s=t'+1}^{t-1} \Phi_{t,s}\Delta A_s^{cl}\hat{\Phi}_{s-1,t'}\big)\hat{B}_{t'}\| \\ &\leq \sum_{t=1}^{T} L \sum_{t'=0}^{t} \big(M\alpha^{t-t'}\Delta + \big(\sum_{s=t'+1}^{t-1} M\alpha^{t-s}\Delta M\alpha^{s-t'}\big)L\big). \end{aligned}$$

Note that the preceding analysis holds for any choice of θ and (x_0, ψ). Thus, noting that

$$\sum_{s=t'+1}^{t-1} M\alpha^{t-s}\Delta M\alpha^{s-t'} < M^2\frac{1}{1-\alpha}\Delta$$

in the case where $\alpha < 1$, leveraging the preceding inequality we can easily conclude that there exists $C > 0$ sufficiently large such that for each θ and (x_0, ψ) we have:

$$\|\nabla_\theta J_T(\theta; x_0, \psi) - \hat{g}_T(\theta; x_0, \psi)\| \leq \begin{cases} CT^3\alpha^T\Delta & \text{if } \alpha > 1 \\ CT^3\Delta & \text{if } \alpha = 1 \\ CT^2\Delta & \text{if } \alpha < 1, \end{cases}$$

which demonstrates the desired result. □

Lemma 5. *Let Assumptions 10-13 hold. Then there exists $K > 0$ independent of $T \in \mathbb{N}$, M and $\alpha \in \mathbb{R}$ such that for each $x_0 \in D$, $\psi \in \Psi$ and $\theta \in \Theta$ we have:*

$$\|\nabla^2_\theta J_T(\theta; x_0, \psi)\| \leq \begin{cases} KT^4\alpha^{3T} & \textit{if } \alpha > 0 \\ KT^4 & \textit{if } \alpha = 0 \\ KT & \textit{if } \alpha < 0. \end{cases}$$

Proof. Let the constant $L > 0$ be large enough so that it upper-bounds the norm of the first and second partial derivatives of Q_t, R_t, π^θ_t, F and $\hat{F}$ on the set $\bar{D} \times \Psi \times \Theta$. Note that the existence of such a constant is stipulated by Assumption 11. Fix a specific task (x_0, ψ) and set of policy parameters θ. Recall from (3.22) that the Hessian can be calculated as follows:

$$\begin{aligned} \nabla^2 J_T(\theta; x_0, \psi) = &\big(\frac{\partial x_T}{\partial \theta}\big)^T \cdot \nabla^2 Q_T(x_T) \cdot \frac{\partial x_T}{\partial \theta} \\ &+ \sum_{t=0}^{T-1} \big(\frac{\partial x_t}{\partial \theta}\big)^T \cdot \frac{\partial^2}{\partial x^2} H_t(x_t, p_t, \theta) \cdot \frac{\partial x_t}{\partial \theta} \\ &+ 2\sum_{t=0}^{T-1} \big(\frac{\partial x_t}{\partial \theta}\big)^T \cdot \frac{\partial^2}{\partial x \partial \theta} H_t(x_t, p_{t+1}, \theta) \\ &+ \sum_{t=0}^{T-1} \frac{\partial^2}{\partial \theta^2} H_t(x_t, p_{t+1}, \theta), \end{aligned}$$

By Assumption 11, and using the form of the Hamiltonian (3.26), we see that there exists a constant $C_1 > 0$ sufficiently large such that

$$\max\{\frac{\partial^2}{\partial x^2} H_t(x_t, p_t, \theta), \frac{\partial^2}{\partial x \partial \theta} H_t(x_t, p_{t+1}, \theta), \frac{\partial^2}{\partial x \partial \theta} H_t(x_t, p_{t+1}, \theta)\} \leq C_1(\|p_{t+1}\| + 1) \quad (3.44)$$

and

$$\|\nabla^2 J_T(\theta; x_0, \psi)\| = L\|\frac{\partial x_T}{\partial \theta}\|^2 + \sum_{t=0}^{T-1} C_1(\|p_{t+1}\| + 1)\big[\|\frac{\partial x_T}{\partial \theta}\|^2 + \|\frac{\partial x_t}{\partial \theta}\| + 1\big] \tag{3.45}$$

holds for all choices of (x_0, ψ) and θ.

Using our preceding analysis, we can bound the derivative as the state trajectory as follows:

$$\begin{aligned}\|\frac{\partial x_t}{\partial \theta}\| &= \|\sum_{t'=0}^{t-1} \Phi_{t,t'} B_{t'} \frac{\partial \pi_t^\theta}{\partial \theta}\| \\ &\leq \sum_{t'=0}^{t-1} L^2 M \alpha^{t-t'}\end{aligned}$$

This demonstrates that there exists $C_2 > 0$ sufficiently large such that:

$$\|\frac{\partial x_t}{\partial \theta}\| \leq \begin{cases} C_2 T \alpha^T & \text{if } \alpha > 1 \\ C_2 T & \text{if } \alpha = 1 \\ C_2 & \text{if } \alpha < 1, \end{cases} \tag{3.46}$$

where in the case where $\alpha < 1$ we have used the fact that $\sum_{t'=0}^{t-1} M\alpha^{t-t'} < M\frac{1}{1-\alpha}$. Combining the previous bounds (3.44), (3.42) and (3.45) then demonstrates the desired result.

□

Chapter 4

Computational Bottlenecks for Nonlinear Optimal Control

The primary appeal of optimal control is that it allows the user to implicitly encode potentially complex stabilizing controllers as the feedback solutions to certain infinite horizon optimal control problems which are relatively simple to specify. In principle, obtaining an optimal infinite horizon controller requires solving the Hamilton-Jacobi-Bellman partial differential equation [9]. However, for general nonlinear problems it is rarely possible to solve the equation in closed form. This has lead to the development of numerous computational methods which approximate the optimal infinite horizon controller, such as dynamic programming [14], receding horizon control [34] and approximate dynamic programming [14] (including modern reinforcement learning methods [35]). Each of these methods has algorithmic parameters which trade off the quality of the approximation with the amount of computation or number of samples from the system that are used to solve the problem.

This chapter asks a basic question: how does the chosen cost function interact with the inherent geometry of the control system to affect the amount of computation needed to obtain a stabilizing controller? To make this question tractable, we consider a two-stage cost design process described below. To concretely characterize the amount of computation needed to obtain a stabilizing controller with a given cost, our theoretical analysis focuses on receding horizon control (RHC) and the dynamic programming algorithm known as value iteration (VI). For these methods we respectively characterize the prediction horizon $T > 0$ and number of iterations $k \in N$ the two methods require to stabilize the system. Prior work has shown a direct relationship between these two quantities [9], thus we primarily analyze RHC schemes and then use these results to characterize VI. On the empirical side, we investigate how the choice of cost function affects the amount of data modern reinforcement learning algorithms need to stabilize the system.

The first step in the cost design process we consider is to select a set of outputs $y = h(x)$ to penalize in the objective, where x is the state of the system. We will then consider running costs of the form $\ell^\epsilon(x, u) = \|h(x)\|_2^2 + \epsilon\|u\|_2^2$, where u is the system input and the choice of

weighting parameter $\epsilon > 0$ represents the second design choice. Intuitively, as $\epsilon > 0$ is made smaller, the running cost encourages controllers which more aggressively drive the outputs to zero. During our analysis, we will first fix a set of outputs and investigate how the choice of $\epsilon > 0$ affects the amount of computation needed to obtain a stabilizing controller.

Our theoretical analysis draws on insights from two distinct lines of work. The first insight comes from the receding horizon literature [33, 30, 63], which relates the optimal infinite horizon performance for a given cost to the prediction horizon needed by RHC schemes (and VI) to stabilize the system. Informally, the smaller the infinite horizon cost, the shorter the prediction horizon $T > 0$ needed to stabilize the system. The rough intuition here [63] is that as the optimal infinite horizon cost becomes smaller the optimal controller must necessarily drive the running cost to zero more rapidly, and this myopic behavior is easier for RHC schemes to approximate with a short prediction horizon.

The second line of work is the 'cheap control' literature [97, 3, 87, 88], which studies the class of cost functions we consider and draws a separation between minimum-phase (MP) and non-minimum-phase (NMP) systems. In particular, this work bounds the performance of the optimal infinite horizon controller as $\epsilon \to 0$. As the limit is taken the optimal controller drives the outputs to zero as rapidly as possible while ensuring the closed-loop system is asymptotically stable. For MP systems, under suitable conditions, the infinite horizon performance can be made arbitrarily small by taking $\epsilon \to 0$, as a high-gain output-zeroing controller will not destabilize the zeros. However, for NMP systems the unstable zero dynamics present a fundamental barrier to making the infinite horizon performance arbitrarily small, as the optimal controller cannot myopically drive the outputs to zero and must instead 'steer' the outputs to stabilize the zeros [97].

We build on these perspectives by demonstrating that when the chosen outputs lead to MP behavior the prediction horizon $T > 0$ needed for RHC schemes to stabilize the system can be made arbitrarily small by making $\epsilon > 0$ sufficiently small. Thus, when the system is MP, the user can consistently decrease the computational burden of obtaining a stabilizing controller by using costs which encourage myopically driving the outputs to zero. In sharp contrast, for NMP systems as we take $\epsilon \to 0$ the prediction horizon needed to stabilize the system actually increases and becomes unbounded. Thus, 'high-performance' infinite horizon optimal controllers which zero the outputs as rapidly as possible are difficult to approximate in the NMP case but not in the MP case (using RHC and VI). Moreover, we identify a class of passively unstable NMP systems for which there is a minimum prediction horizon $T > 0$ that is needed to stabilize the system *for all choices of* $\epsilon > 0$. Taken together, these results demonstrate that NMP dynamics constitute an obstacle, from a computational perspective, to constructing a stabilizing controller. Our simulation studies with reinforcement learning further support this perspective.

Cost Formulation

For each $\epsilon > 0$ and $T > 0$ we study the cost functions:

$$\inf_{u(\cdot)\in U_\infty} J^\epsilon_\infty(u(\cdot); x_0) := \int_0^\infty \|h(x(t))\|_2^2 + \epsilon\|u(t)\|_2^2 dt, \tag{4.1}$$

$$\inf_{u(\cdot)\in U_T} J^\epsilon_T(u(\cdot); x_0) := \int_0^T \|h(x(t))\|_2^2 + \epsilon\|u(t)\|_2^2 dt, \tag{4.2}$$

where $h\colon \mathbb{R}^n \to \mathbb{R}^p$ is a map constructed by the user, which is assumed to be twice continuously differentiable and such that $h(0) = 0$. The map h captures the physical quantities of the system which are the most important to drive to zero to meet the control objectives of the designer.

To each of these problems we associate value functions:

$$V^\epsilon_\infty(x_0) := \inf_{u\in U_\infty} J^\epsilon_\infty(u(\cdot); x_0), \tag{4.3}$$

$$V^\epsilon_T(x_0) := \inf_{u\in U_T} J^\epsilon_T(u(\cdot); x_0). \tag{4.4}$$

We assume that for each $\epsilon > 0$ V^ϵ_∞ is positive definite, continuous and bounded on bounded sets so that the infinite horizon controller will stabilize the system [9].

Normal Forms, Zero Dynamics and Structural Assumptions

To better understand how the cost functions introduced in the previous section interact with the geometry of the state-space model (4.16), let us formally append a set of outputs to the system and form an input-output model of the form:

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x), \end{aligned} \tag{4.5}$$

where $y \in \mathbb{R}^p$. As alluded to above, the perspective here is that the choice of the running cost *implicitly induces this structure*, and our goal throughout the chapter is to understand how this choice impacts the design trade-offs available to the user. Thus, in this section we briefly review basic concepts from nonlinear geometric control which shed light onto how the choice of cost function interacts with the underlying structure of the dynamics. In particular, we discuss how to construct a 'normal form' associated to the input-output system (4.5), and also introduce several simplifying assumption we will make throughout the chapter. Our introduction to these concepts will be brief, as they are covered in many standard references (e.g. [90, Chapter 9]). We first make the following Assumption:

Assumption 14. *The number of inputs is equal to the number of outputs, namely, $q = p$.*

We make this assumption as the construction of normal forms is more straightforward for 'square' systems. We discuss the impact of removing this structural condition, along with Assumptions 15 and 16 below, in Section 4.2.

To obtain a more direct expression relating the evolution of the outputs to the inputs, one can repeatedly differentiate each of the outputs along the dynamics (4.16) until an expression of the following form is obtained:

$$\begin{bmatrix} y_1^{(r_1)} & \dots & y_q^{(r_q)} \end{bmatrix}^T = b(x) + A(x)u, \tag{4.6}$$

where $y_j^{(k)}$ denotes the k-th time derivative of $y_j = h_j(x)$ (the j-th entry of the output) and the r_j are positive integers. If the matrix $A(x)$ is bounded away from singularity for each $x \in \mathbb{R}^n$ then we say that the system has a well-defined (vector) relative degree $\bar{r} = (r_1, r_2, \dots, r_q)$. In this case there exists a valid change of coordinates $x \to (\xi, \eta)$ such that in the new coordinates the dynamics are of the form:

$$\begin{aligned} \dot{\xi} &= F\xi + G\left[\bar{b}(\xi,\eta) + \bar{A}(\xi,\eta)u\right] \\ \dot{\eta} &= \bar{q}(\xi,\eta) + \bar{P}(\xi,\eta)u \\ y &= C\xi, \end{aligned} \tag{4.7}$$

where (F, G) is controllable, (F, C) is observable and $\bar{A}(\xi, \eta)$ is bounded away from singularity for each $(\xi, \eta) \in \mathbb{R}^n$. Here the coordinates $\xi \in \mathbb{R}^{|\bar{r}|}$ capture the outputs and dervatives up to the appropriate order and $\eta \in \mathbb{R}^{n-|\bar{r}|}$ completes the change of coordinates. Namely, ξ contains entries of the form $\xi_{j,k} = y_j^{(k-1)}$ for $j = 1, \dots, q$ and $k = 1, \dots, r_j$.

Next we discuss two simplifying assumption we will use, which are in line with the cheap control literature [17]:

Assumption 15. *There exists $r \in \mathbb{N}$ such that the vector relative degree of the system (4.16) is $\bar{r} = (r, r, \dots, r)$.*

Under assumption 15, we can arrange $\xi = (\xi_1, \xi_2, \dots \xi_r)$ and the F, G and C matrices in (4.7) to be of the form:

$$F = \begin{bmatrix} 0 & I & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & I \\ 0 & 0 & \dots & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ I \end{bmatrix}, C = \begin{bmatrix} I & 0 & \dots & 0 \end{bmatrix}. \tag{4.8}$$

Namely, $\xi_1 = (y_1, \dots, y_q)$ represents the outputs and for $k = 2, \dots, r$ the coordinates $\xi_k = (y_1^{(k-1)}, \dots, y_q^{(k-1)})$ capture the $(k-1) - th$ derivatives of the outputs.

Next, we will restrict the structure of the interconnection between the ξ and η subsystems. We say that the input-output system (4.16) can be put into *strict feedback form* if the η coordinates can be chosen so that the normal form of the dynamics takes the following form:

$$\begin{aligned} \dot{\xi} &= F\xi + G[\bar{b}(\xi,\eta) + \bar{A}(\xi,\eta)u] \\ \dot{\eta} &= \bar{f}_0(\eta) + \bar{g}_0(\eta)\xi_1 \\ y &= \xi_1 = C\xi, \end{aligned} \tag{4.9}$$

where $\bar{f}_0 \colon \mathbb{R}^{n-|\bar{r}|} \to \mathbb{R}^{n-|\bar{r}|}$ and $\bar{g}_0 \colon \mathbb{R}^{n-|\bar{r}|} \to \mathbb{R}^{(n-|\bar{r}|)\times q}$.

Assumption 16. *The input-output system* (4.16) *can be put into the strict feedback form* (4.9).

This assumption forbids the input and the derivatives of the outputs to appear directly in the dynamics of the zeros, and prevents the so-called *peaking phenomena* [96], which we discuss in more detail in Section 4.2. In the special case of linear dynamics the system can always be put into strict feedback form when Assumption 15 is satisfied (see e.g. [88]).

For this class of system the *zero dynamics* are obtained by setting the outputs to zero:

$$\dot{\eta} = \bar{f}_0(\eta). \tag{4.10}$$

We say that the input-output system is *minimum-phase* (MP) if the zero dynamics are asymptotically stable, and *exponentially minimum-phase* if they are exponentially stable. We say that this system is *non-minimum-phase* (NMP) if it is not minumum-phase, and *exponentially non-minimum-phase* if the dynamics $\dot{\eta} = -\bar{f}_0(\eta)$ are exponentially stable. Finally, if $|\bar{r}| = n$ then no η coordinates are needed in the coordinate transformation, and the system is called *full-sate linearizable*. By way of convention, systems which are full-state linearizable are trivially (exponentially) minimum-phase.

The Cheap Control Limit

The focus of the cheap control literature has been to characterize the structure of the optimal value function V_∞^ϵ and corresponding optimal controller u^ϵ for small values of $\epsilon > 0$. In particular, the limiting value $\lim_{\epsilon\to 0} V_\infty^\epsilon(x)$ provides qualitative insight into how difficult it is to drive the chosen outputs to zero from the state $x \in \mathbb{R}^n$ while also stabilizing the internal dynamics. The essential result from the literature is a qualitative separation between the performance limitations of MP and NMP systems. While the majority of the literature has focused on the case where the dynamics are linear [87, 88, 27], [97] and [17] extend these results to nonlinear strict-feedback systems of the form (4.9). An integral part of the analysis for strict feedback systems is the 'minimum energy problem' which is formulated using the normal form (4.9):

$$\hat{V}_0(\eta_0) = \inf_{\xi_1(\cdot)} J_0(\xi_1(\cdot);\eta_0) = \int_0^\infty \|\xi_1(t)\|_2^2 dt \tag{4.11}$$

where $\dot{\eta} = f_0(\eta) + g_0(\eta)\xi_1$, the output $\xi_1(\cdot)$ is viewed as an 'input' to the zero subsystem and the infimum in (4.11) is understood to be over $\xi_1(\cdot)$ which drive $\eta(t) \to 0$ asymptotically. Thus, $\hat{V}_0(\eta)$ can be interpreted as the minimum 'energy' of the outputs (in an $\mathcal{L}_2$ sense) that must be accrued by a feedback controller which stabilizes the internal dynamics. When $\hat{V}_0$ is continuously differentiable, an 'optimal controller' for the zeros subsystem is given by $\xi_1(t) = \mu_0(\eta(t))$, where $\mu_0 \colon \mathbb{R}^{n-|\bar{r}|} \to \mathbb{R}^q$ is obtain from the HJB PDE associated to the reduced problem (4.11).

Crucially, one may observe that if the system is MP then $\hat{V}_0(\cdot) \equiv 0$ since no 'energy' must be expended to stabilize the zeros. On the other hand, when the system is NMP we will have $\hat{V}_0(\cdot) \not\equiv 0$ since the outputs must be 'steered' to stabilize the zeros. In both cases, under suitable technical conditions, the performance limitation for the system is given by $\lim_{\epsilon \to 0} \hat{V}^{\epsilon}_{\infty}(\xi, \eta) = \hat{V}_0(\eta)$, where $\hat{V}^{\epsilon}_{\infty}(\xi, \eta)$ is the representation of the value function in the normal coordinates. Thus, for MP systems the infinite horizon cost can be made arbitarily small by taking $\epsilon \to 0$, while there is a fundamental limitation for NMP systems. For MP systems as $\epsilon \to 0$ the optimal controller drives the outputs directly to zero more and more rapidly, while in the NMP case a high-gain feedback controller drives $\xi(t) \to \mu_0(\eta(t))$ to stabilize the zeros [17].

Fast-Slow Representations

As mentioned above, singular perturbation techniques play a crucial role in obtaining the aforementioned results and play an essential role in our analysis. Even though most of our arguments are relegated to the supporting document [102], it is worthwhile to outline the broad strokes of the analysis here.

In particular, first define the new parameter $\tilde{\epsilon} > 0$ such that $\epsilon = \tilde{\epsilon}^{2r}$ so that we may rewrite the running as $\|\xi_1\|_2^2 + \tilde{\epsilon}^{2r}\|u\|_2^2$. If we then define the new coordinates

$$\tilde{\xi} = S(\tilde{\epsilon})\xi \qquad \text{and} \quad \tilde{u} = \tilde{\epsilon}^r u, \tag{4.12}$$

where

$$S(\tilde{\epsilon}) = \operatorname{diag}(1, \tilde{\epsilon}, ..., \tilde{\epsilon}^{r-1}) \tag{4.13}$$

then the system (4.9) takes on the fast-slow representation:

$$\begin{aligned} \tilde{\epsilon}\dot{\tilde{\xi}} &= F\tilde{\xi} + G\big[\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}, \eta) + \tilde{A}(\tilde{\xi}, \eta)\tilde{u}\big] \\ \dot{\eta} &= \bar{f}_0(\eta) + \bar{g}_0(\eta)\tilde{\xi}_1, \end{aligned} \tag{4.14}$$

where $\tilde{b}(\tilde{\xi}, \eta) = \bar{b}(S(\tilde{\epsilon})^{-1}\tilde{\xi}, \eta)$ and $\tilde{A}(\tilde{\xi}, \eta) = \bar{A}(S(\tilde{\epsilon})^{-1}\tilde{\xi}, \eta)$ and we have suppressed the dependence of these terms on $\tilde{\epsilon}$. In the re-scaled coordinates the infinite horizon cost becomes:

$$\inf_{\tilde{u}(\cdot) \in U_{\infty}} \tilde{J}^{\tilde{\epsilon}}_{\infty}(\tilde{u}(\cdot); \tilde{\xi}_0, \eta_0) = \int_0^{\infty} \|\tilde{\xi}_1(t)\|_2^2 + \|\tilde{u}(t)\|_2^2 dt. \tag{4.15}$$

We will let $\tilde{V}_\infty^{\tilde{\epsilon}}$ denote the representation of the value function in the new coordinates, and we define the reparameterized finite horizon cost $\tilde{J}_T^{\tilde{\epsilon}}$ and optimal performance $\tilde{V}_T^{\tilde{\epsilon}}$ in an analogous way. The form of the re-scaled cost function and dynamics clearly evokes the intuition that we should expect a fast transient for the outputs for small values of $\tilde{\epsilon} > 0$.

4.1 Infinite Horizon Optimal Control, Receding Horizon Approximations, And Value Iteration

We will consider a system of the form:

$$\dot{x} = f(x) + g(x)u, \qquad x(0) = x_0, \tag{4.16}$$

where $x \in \mathbb{R}^n$ is the state, $x_0 \in \mathbb{R}^n$ is the initial condition and $u \in \mathbb{R}^q$ is the input. We will assume that the maps $f\colon \mathbb{R}^n \to \mathbb{R}$ and $g\colon \mathbb{R}^n \to \mathbb{R}^{n\times q}$ are twice continuously differentiable and that $f(0) = 0$. For each $T \in \mathbb{R} \cup \{\infty\}$ we will let U_T denote the set of controls of the form $u\colon [0, T] \to \mathbb{R}^q$ which are measurable and essentially bounded.

Infinite Horizon Optimal Control

In this section we will consider a generic infinite horizon optimal control problem of the form:

$$\inf_{u(\cdot)\in U_\infty} J_\infty(u(\cdot); x_0) := \int_0^\infty \ell(x(t), u(t))dt, \tag{4.17}$$

where $(x(\cdot), u(\cdot))$ are subject to (4.16) and $\ell\colon \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}$ is a twice continuously differentiable running cost which is non-negative, strictly convex in u, and satisfies $\ell(0, 0) = 0$. To the infinite horizon cost we associate the value function:

$$V_\infty(x_0) := \inf_{u(\cdot)\in U_\infty} J_\infty(u(\cdot); x_0).$$

We assume that V_∞ is continuous, positive definite, bounded on bounded sets, and that there exists a control which achieves the optimal performance, namely, $V_\infty(x_0) = J_\infty(u^*(\cdot); x_0)$ for some $u^* \in U_\infty$. We will implicitly make these standard assumptions for each of the costs used later in the chapter.

It is well-known that V_∞ can be obtained, in principle, as a solution to the Hamilton-Jacobi-Bellman equation (see e.g. [9, Chapter 3.2]) and used to synthesize an optimal stabilizing feedback controller $u_\infty\colon \mathbb{R}^n \to \mathbb{R}^q$ for the cost. This controller is optimal in the sense that when applied to the system it achieves the smallest possible cost from each initial condition.

Receding Horizon Control

Next, we discuss receding horizon approximations to the optimal infinite horizon controller u_∞. These schemes use a finite horizon approximation to (4.17) of the form:

$$\inf_{u(\cdot)\in U_T} J_T(u(\cdot);x_0) = \int_0^T \ell(x(t),u(t))dt, \tag{4.18}$$

where $T>0$ is a finite prediction horizon. The value function associated to the approximation is:

$$V_T(x_0) := \inf_{u(\cdot)\in U_T} J_T(u(\cdot);x_0).$$

To ease exposition we assume that for each $T>0$ and $x_0\in\mathbb{R}^n$ there exists a unique minimizer $u_T^*(\cdot;x_0)\in U_T$ for the optimal control problem, and we will let $x_T^*(\cdot;x_0)$ denote the corresponding state trajectory. However, we note that in the case where there are multiple optimal controls for a given initial condition the following discussion goes through if any of these control signals are used.

For each prediction horizon $T>0$ and sampling rate $T\geq\Delta t>0$, receding horizon schemes approximate the infinite horizon controller u_∞ with a sampled data control law of the form $u_{T,\Delta t}(t;x_0)=u_T^*(t-t_k;x_{T,\Delta t}(t_k;x_0))$ for each $t\in[t_k,t_{k+1})$, where the $t_k=k\Delta t$ are sampling instances and $x_{T,\Delta t}(\cdot;x_0)$ is the state trajectory produced by the receding horizon scheme from the initial condition $x_0\in\mathbb{R}^n$. In words, at each sampling instant t_k the receding horizon controller solves the finite horizon optimal control problem (4.18) from the current system state, applies the resulting open loop control for Δt seconds, and then repeats the process at time t_{k+1}.

The Stability of RHC

At their core, stability results from the literature [42, 34] are founded on the notion that as $T>0$ increases the RHC scheme more closely approximates the infinite horizon continuous-time feedback controller u_∞. The quality of this approximation is typically characterized by how the values of V_T converge to those of V_∞. However, increasing T comes at the cost of additional computational complexity when solving (4.18). We next describe a specific stability result used in our analysis, which upper-bounds the prediction horizon $T>0$ needed to ensure stability. In what follows, we will let $\sigma\colon\mathbb{R}^n\to\mathbb{R}$ be a fixed positive definite function which will be used to measure the distance of the state to the origin.

Assumption 17. *There exists $\bar{\alpha}_V>0$ such that:*

$$V_\infty(x)\leq\bar{\alpha}_V\cdot\sigma(x)\qquad\forall x\in\mathbb{R}^n.$$

Assumption 18. *There exists a continuously differentiable function $W\colon\mathbb{R}^n\to\mathbb{R}$ and $\bar{\alpha}_W,\underline{\alpha}_W,K_W>0$ such that for each $x\in\mathbb{R}^n$ and $u\in\mathbb{R}^q$:*

$$\underline{\alpha}_W\cdot\sigma(x)\leq W(x)\leq\bar{\alpha}_W\cdot\sigma(x)$$

$$\frac{d}{dx}W(x)[f(x)+g(x)u] \leq -K_W \cdot \sigma(x) + \ell(x,u).$$

Intuitively, the smaller the constant $\bar{\alpha}_V > 0$ in Assumption 17 the more rapidly the infinite horizon optimal controller must drive the instantaneous performance to zero. The existence of the map W in Assumption 18 ensures that the state measure $\sigma(\cdot)$ is detectable with respect to the loss function ℓ, in the sense that if $\ell(x,u) = 0$ then we must have $\frac{d}{dt}W(x) < 0$ if $x \neq 0$. The following result, which we prove in the supplementary document [102], is essentially a continuous-time adaptation and specialization of the stability result from [34] which is stated for discrete-time receding horizon control:

Theorem 4. *Let Assumptions 17 and 18 hold. Then the receding horizon controller $u_{T,\Delta t}$ will asymptotically stabilize* (4.16) *for each for each $T \geq \Delta t \geq 0$ such that:*

$$T > \frac{\bar{\alpha}_V(\bar{\alpha}_V + \bar{\alpha}_W)}{K_W \underline{\alpha}_W (1 - M(\Delta t))} \tag{4.19}$$

where

$$M(\Delta t) = \exp(-K_W \frac{\underline{\alpha}_W \Delta t}{\bar{\alpha}_V + \bar{\alpha}_W})] < 1. \tag{4.20}$$

Note how the bound on the required prediction horizon $T > 0$ depends on the performance of the infinite horizon cost. As $\bar{\alpha}_V$ decreases we can ensure asymptotic stability of the closed-loop system by using RHC schemes with smaller and smaller prediction horizons. This will form the basis for our stability results for minimum-phase systems, when combined with the cheap control results in Section 4.

Connections Between RHC and VI

Again using a sampling interval $\Delta t > 0$, the value iteration (VI) algorithm constructs a sequence of approximations $V^1, V_2, \ldots$ to the infinite horizon value function V_∞ using the following recursion with $V^1(\cdot) \equiv 0$:

$$V^{k+1}(x_0) = \inf_{u \in U_{\Delta t}} \left[\int_0^{\Delta t} \ell(x(t), u(t))dt + V^k(x(\Delta t)) \right]. \tag{4.21}$$

For each $k \in \mathbb{N}$ let $u^k(\cdot; x_0) \in U_{\Delta t}$ be a control which solves the optimization on the right-hand side of (4.21). Under mild conditions one can show that V^k converges to V_∞ as $k \to \infty$ (see [9, Chapter 3.3] for a more in depth discussion). The algorithm produces a sampled-data control law of the form $u^{k,\Delta t}(t; x_0) = u^k(t - t_k; x^{k,\Delta t}(t_k))$ for each $t \in [t_k, t_{k+1})$ where $x^{k,\Delta t}(\cdot; x_0)$ is the state trajectory produced by the control scheme from the given initial condition $x_0 \in \mathbb{R}^n$.

Using the Principle of Optimality (see e.g. [9, Prop 3.2]), one can show that for each $k \in \mathbb{N}$ VI produces the estimate $V^k = V_{k \cdot \Delta t}$. Moreover the k-th greedy control is characterized for

each $x_0 \in \mathbb{R}^n$ by $u^k(\cdot; x_0) = u_T(\cdot; x_0)|_{[0,\Delta t]}$. Thus, the sampled data controller obtained after k steps of VI with discretization parameter Δt is equivalent to the receding horizon controller with prediction horizon $T = k\Delta t$ and re-planning interval Δt. Thus, the dynamic programming based controller $u^{k,\Delta t}$ implicitly optimizes over the prediction horizon $T = k\Delta t$, and there is a direct correspondence between cases where RHC and VI will stabilize (4.16).

4.2 The Computational Consequences of Cost Design for Nonlinear Optimal Control

We are now ready to present our theoretical results and draw a qualitative distinction between what is possible, from a computational perspective, when the outputs $y = h(x)$ correspond to either minimum-phase or non-minimum-phase behavior. Due to space constraints, proofs of the following results are relegated to [102] and we only outline the main arguments here. In Section 4.2 we introduce bounds on V_∞^ϵ and V_T^ϵ which are used in the proofs of the main results and provide qualitative insight into how well receding horizon controllers approximate u_∞^ϵ. We then discuss our stability result for minimum-phase systems in Section 4.2 and instability results for non-minimum-phase system in Section 4.2.

Performance Bounds

Our results require the following growth assumptions:

Assumption 19. *There exists $C > 0$ such that the following conditons hold for each $x \in \mathbb{R}^n$ and $(\xi, \eta) \in \mathbb{R}^n$:*

$$\begin{aligned} &\|f(x)\|_2 \le L\|x\|_2, && \|\bar{b}(\xi,\eta)\|_2 \le L\left(\|\xi\|_2 + \|\eta\|_2\right), \\ &\|g(x)\|_2 \le L, && \|\bar{A}(\xi,\eta)\|_2 < L, \\ &\|\bar{f}_0(\eta)\|_2 \le L\|\eta\|_2, && \|\bar{g}_0(\eta)\|_2 \le L. \end{aligned}$$

Under this regularity condition we can obtain the following bound on the infinite horizon cost for MP systems:

Lemma 6. *Let Assumptions 14-19 hold. Further assume that* (4.9) *is exponentially minimum-phase (including full-state linearizable). Then there exist $\hat{K} > 0$ such that for each $0 < \tilde{\epsilon} \le 1$ we have for each $(\tilde{\xi}, \eta) \in \mathbb{R}^n$:*

$$\tilde{V}_\infty^{\tilde{\epsilon}}(\tilde{\xi}, \eta) \le \hat{K}\big(\tilde{\epsilon}\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}^{2r}\|\eta\|_2^2\big). \tag{4.22}$$

The proof uses the fast-slow representation of the dynamics (4.14) and bounds the infinite horizon performance of a sub-optimal feedback linearizing controller of the form $\tilde{u} = \tilde{A}^{-1}(\tilde{\xi}, \eta)[-\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}, \eta) + K\tilde{\xi}]$ where $F + GK$ is Hurwitz, which drives the $\tilde{\xi}$ coordinates to zero exponentially at a rate on the order of $\frac{1}{\tilde{\epsilon}}$. Because the zero dynamics are exponentially

minimum-phase, and we have restricted the interconnection between the two systems with Assumption 16, this high gain feedback does not destabilize the zeros. As expected, Lemma 6 implies that the infinite horizon optimal controller drives the outputs to zero more and more rapidly as $\tilde{\epsilon} \to 0$. In contrast, we recall from Section 4 that $\tilde{V}_{\infty}^{\tilde{\epsilon}}$ can be lower-bounded uniformly in the case where the system is NMP, since the outputs must be 'steered' so as to stabilize the zeros.

Next, we discuss a bound on the finite-horizon performance which holds for both MP and NMP systems:

Lemma 7. *Let Assumptions 14-19 hold. Then for each $\bar{T} > 0$, there exists $\bar{K} > 0$ and $\bar{\epsilon} > 0$ such that for each $\bar{T} \geq T > 0$ and $\epsilon \in (0, \bar{\epsilon}]$ we have for each $(\tilde{\xi}, \eta) \in \mathbb{R}^n$:*

$$\tilde{V}_T^{\tilde{\epsilon}}(\tilde{\xi}, \eta) \leq \bar{K}\big(\tilde{\epsilon}\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}^{2r}\|\eta\|_2^2\big). \tag{4.23}$$

The result is again obtained by bounding the performance of a linearizing controller which drives the $\tilde{\xi}$ coordinates to zero. Unlike in the infinite horizon case, on bounded time horizons the optimal control does not need to drive the zeros to the origin to achieve a finite cost, which vanishes as $\tilde{\epsilon} \to 0$. Indeed, as the preceding bound indicates, and as we show more formally in the proof of Theorem 7 in [102], when the prediction horizon $T > 0$ is bounded and $\tilde{\epsilon} > 0$ is small the optimal control always drives the outputs toward zero in a mypoic fashion. In the NMP case, this will mean that RHC schemes fails to stabilize the zero dynamics when $\tilde{\epsilon}$ is small, unless a very large prediction horizon is used.

We provided the previous bound in terms of the rescaled coordinates $(\tilde{\xi}, \eta)$ and the parameter $\tilde{\epsilon} = \epsilon^{\frac{1}{2r}}$, as doing so cleanly separates how the bound depends on the outputs (and their derivatives) and the zeros. We note that in the original representation the bounds on $V_T^{\epsilon}(x)$ and $V_{\infty}^{\epsilon}(x)$ will both be on the order of $O(\epsilon^{\frac{1}{2r}})$, providing insight into how the relative degree affects the growth of the bound. We state our main results below using the original representation for the problem.

Stability Results for Full-State Linearizable Systems

In this Section we provide our cost-shaping result for cases where the chosen outputs induce an input-output system which is full-state linearizable. For the following result we are able to apply the argument from 4. Even though this result is really a corollary to Theorem 6 presented below, as it is instructive to compare the two proof techniques to see how 'standard' stability results from the RHC literature are unfit for the more general minimum-phase case considered later. We will provide commentary on how to interpret this result (as a special case of the general MP result in 6) after introducing both results.

Lemma 8. *Let Assumptions 15, 16 and 19 hold. Further assume that the system is full-state linearizable. Then for each $T \geq \Delta t > 0$ there exists $\epsilon_0 > 0$ such that for each $\epsilon \in (0, \epsilon_0]$ the corresponding receding horizon controller renders the closed-loop system globally exponentially stable.*

Theorem 5. *Let Assumptions 14-19 hold. Further assume that* (4.9) *is full stae linearizable. Then for every fixed $T \geq \Delta t > 0$ there exists $\bar{\epsilon} > 0$ such that for each $\epsilon \in (0, \bar{\epsilon}]$ the receding horizon controller $u^{\epsilon}_{T,\Delta t}(\cdot; x_0)$ renders the closed-loop system globally exponentially stable.*

Proof. The proof will work in the $\tilde{\xi}$ coordinates and uses the storage function $W(\tilde{\xi}) = \tilde{\xi} P \tilde{\xi}$ where P solves the Lyapunov equation $(F + MH)^T P + P(F + MH) = -2I$ where I is the identity, M is chosen so that $(F + MH)$ is Hurwitz (which is feasible since (F, H) is detectable), and all matrices are of appropriate dimension. Our choice for W is inspired by the construction in [34], which instead studies discrete time linear systems. With an eye towards the satisfaction of Assumption 18, we can compute:

$$\begin{aligned}
\dot{W}^{\epsilon}(\tilde{\xi}, \tilde{u}) &= \tilde{\xi}^T (F^T P + PF)\xi + 2\tilde{\xi}^T PG\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}) + 2\tilde{\xi}^T PG\tilde{A}(\tilde{\xi})u \\
&= \tilde{\xi}^T ((F + MH)^T P + P(F + MH))\xi - \tilde{\xi}^T (MHP + PMH)\tilde{\xi} + 2\tilde{\xi}^T PG\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}) + 2\tilde{\xi}^T PG\tilde{A}(\tilde{\xi})u \\
&\leq -2\|\tilde{\xi}\|_2^2 + 2\|P\| \cdot \|M \cdot \|\|\tilde{\xi}_1\|_2^2 + 2L \cdot \|P\| \cdot \|G\| \cdot \tilde{\epsilon}^r \cdot \|\tilde{\xi}\|^2 + 2L\|P\| \cdot \|G\| \cdot \|\tilde{\xi}\| \cdot \|\tilde{u}\| \\
&\leq -(1 - 2L \cdot \|P\| \cdot \|G\| \cdot \tilde{\epsilon}^r)\|\tilde{\xi}\|^2 + 2L\|P\| \cdot \|G\| \cdot \|\tilde{\xi}\|(\|\tilde{\xi}_1\|^2 + \|\tilde{u}\|^2) \\
&\leq -(1 - \alpha\tilde{\epsilon}^r)\|\tilde{\xi}\|^2 + \alpha\|\tilde{\xi}\|(\|\tilde{\xi}_1\|^2 + \|\tilde{u}\|^2),
\end{aligned}$$

where we have defined:

$$\alpha = 2L\|P\| \cdot \|G\|. \tag{4.24}$$

From here on let us suppose that $\tilde{\epsilon}^r < \frac{1}{2}\alpha$. Under this condition we have that:

$$\dot{W}(\tilde{\xi}, \tilde{u}) \leq -\frac{1}{2}\alpha\|\tilde{\xi}\| + \alpha\big(\|\tilde{\xi}_1\| + \|\tilde{u}\|^2\big). \tag{4.25}$$

We would like to uses the function W to apply Theorem 4. However, in order to do so we must have $\alpha < 1$. Since this may not be the case in general, we instead consider the rescaling $W \to \frac{1}{\alpha}W$. The preceding inequalities then yields:

$$\frac{1}{\alpha}W(\tilde{\xi}, \tilde{u}) \leq -\frac{1}{2}\|\tilde{\xi}\|^2 + \|\tilde{\xi}_1\| + \|\tilde{u}\|^2, \tag{4.26}$$

which demonstrates that $\frac{1}{\alpha}W$ satisfies the detectability for the running cost in the $\tilde{\xi}, \tilde{u}$ coordinates, namely, in these coordinates the cost satisfies Assumption 18. In particular, $\frac{1}{\alpha}W$ satisfies Assumption 18 with constants $\underline{\alpha}_w = \tilde{\epsilon}\alpha\lambda_{min}(P)$, $\overline{\alpha}_w = \tilde{\epsilon}\alpha\lambda_{max}(P)$ and $K_W = \frac{1}{2}$. Thus, applying Theorem 4

$$T > \frac{\tilde{\epsilon}K(K + \alpha\lambda_{max}(P))}{\frac{1}{2}\alpha\lambda_{min}(P)(1 - M(\Delta t))} \tag{4.27}$$

where

$$M(\Delta t) = \exp\left(-\frac{1}{2}\frac{\alpha\lambda_{min}(P)\Delta t}{K + \alpha\lambda_{\max}(P)}\right) < 1. \tag{4.28}$$

From the preceding inequalities and discussion, we obtain the desired result, as they demonstrate that the system will be stabilized if

$$\tilde{\epsilon}^{1/2} < \tilde{\epsilon} \min\{C(\Delta t)T, \frac{1}{2}\alpha\} \tag{4.29}$$

where

$$C(\Delta t) = \frac{\alpha \lambda_{min}(P)\Delta t}{\hat{K}(\hat{K} + \alpha \lambda_{\max}(P))} \tag{4.30}$$

and the constant $\hat{K} > 0$ is from the upper-bound on the infinite horizon cost in Lemma 7. □

Stability Results and Design Trade-offs for MP Systems

We are ready to state our main result for MP systems, which applies to the case where the zero dynamics are stable, and demonstrates that the prediction horizon $T > 0$ and number of iterations $k \in \mathbb{N}$ required for RHC methods and VI to stabilize the system can respectively be made as small as desired. The result is as follows:

Theorem 6. *Let Assumptions 14-19 hold. Further assume that* (4.9) *is exponentially minimum-phase (including full state-linearizable). Then for every fixed $T \geq \Delta t > 0$ there exists $\bar{\epsilon} > 0$ such that for each $\epsilon \in (0, \bar{\epsilon}]$ the receding horizon controller $u^{\epsilon}_{T,\Delta t}(\cdot; x_0)$ renders the closed-loop system globally exponentially stable.*

The full proof can be found in the Appendix, and we provide some discussion before outlining how the proof diverges from the the fully-linearizable result presented above in Theorem 5.

In the general exponentially MP case the designer can consistently decrease the amount of computation needed to obtain a stabilizing RHC controller (as measured by the prediction horizon $T > 0$ or number of of iterations $k \in \mathbb{N}$) by decreasing $\epsilon > 0$ and encouraging the controller to rapidly drive the outputs to zero. In some applications a fast transient for the outputs may be desirable, and there is no tension between meeting the desired performance objectives and the computational burden of the RHC schemes. In other scenarios the high-gain RHC controllers corresponding to small values of $\epsilon > 0$ may cause undesirable effects such as chattering or use too much input to meet design specifications. In either case, when the chosen outputs are MP, the designer retains the freedom to fully explore these design trade-offs. As we shall see below, design choices become more restricted when the chosen cost functions implicitly induces NMP behavior.

We leave the full proof to the Appendix, but remark here how it differs significantly from the proof of Theorem 5. Working in the $(\tilde{\xi}, \eta)$ coordinates, the main idea behind the proof is to construct a function $W = \tilde{\epsilon} V_1(\tilde{\xi}) + V_2(\eta)$. Here the first function is for the form $V_1(\tilde{\xi}) = \tilde{\xi}^T P \tilde{\xi}$ where P solves the Lyapunov equation $(F + MH)^T P + P(F + MH) = -2I$ where I is the identity, M is chosen so that $(F + MH)$ is Hurwitz (which is feasible since

(F, H) is detectable), and all matrices are of appropriate dimension. This choice of candidate storage function is inspired by the linear systems example from [34]. The second functions V_2 is constructed using a standard converse exponentially stability results [90]. This composite function W can can be shown to satisfy the detectability condition 18. Thus, one might hope that in this case we could again directly apply Theorem 4, along with the performance bound 6 to obtain the desired result. However, the $\tilde{\epsilon} > 0$ scaling factor in the definition of $V_1(\tilde{\xi})$ means that at most we may choose the constant $\underline{\alpha}_W > 0$ to be at most on the order of $\tilde{\epsilon} > 0$, which prevents us from using 4 to argue that we can make $T > 0$ (and consequently $k \in \mathbb{N}$) as small as desired by making $\tilde{\epsilon}$ sufficiently small. This is a reflection of the fact that Theorem 4, and similar arguments that can be found in the literature, are ill-suited for the two time-scale structure that is induced form MP systems which are not full-state linearizable. Thus, the proof of the following results, which can be found in the Appendix, using direct singular perturbation analysis to 1) argue that when $\tilde{\epsilon} > 0$ is small the $\tilde{\xi}$ coordinates are driven exponentially to the origin with a rate of convergence which is on the order of $O(\frac{1}{\epsilon})$ and 2) that the exponential stability of the zeros, under the strict-feedback interconnection Assumption in 16, is not disturbed too much by this fast transient.

Instability Results, Design Trade-offs and Fundamental Limitations for NMP Systems

Our main result for NMP systems in Theorem 7 below highlights a class of NMP systems for which there exists a uniform lower-bound on the prediction horizon $T > 0$ required to stabilize the system with receding horizon methods which holds for all choices of $\epsilon > 0$. We first introduce supportive Lemmas which provide some intuition for this result, and highlight cases when the closed-loop system induced by RHC and VI will be unstable. We note that these results and proof techniques are fairly distinct from previous analysis for RHC and VI presented in the literature. This is unsurprising, as the focus of the literature has naturally been to provide sufficient conditions for stability. n our case the following instability results allow us to draw a qualitative separation between cases where the cost function induces MP vs. NMP dynamics Once again, detailed proofs are relegated to the Appendix. The following result applies to 'small' values of $\tilde{\epsilon}$:

Lemma 9. *Let Assumptions 14-19 hold. Further assume that the system* (4.5) *is globally exponentially NMP. Then for each $\bar{T} > 0$ there exists $\bar{\epsilon} > 0$ such that for each $\epsilon \in (0, \bar{\epsilon}]$ and $\bar{T} \geq T \geq \Delta t > 0$ the receding horizon controller $u^{\epsilon}_{T,\Delta t}$ fails to stabilize* (4.16)*.*

Similarly to the MP case above, the first step in the proof is to argue that for small a fixed $T > 0$ (or $k \in \mathbb{N}$) if $\epsilon > 0$ is sufficiently small then the closed-loop system will once again mypopically drive the outputs and their derivatives to the origin. However in the NMP case, when the outputs are zeroed myopically the zeros will remain unstable. In sharp contrast to the MP case discussed above, Lemma 9 indicates that in the NMP case as we take $\epsilon \to 0$ the time horizon needed for RHC schemes to stabilize the system actually increases

and becomes unbounded. In other words, as $\epsilon \to 0$ and the optimal stabilizing controller u^{ϵ}_{∞} pushes up against the inherent performance limitations of the system, it becomes more difficult to approximate u^{ϵ}_{∞} with receding horizon schemes (and thus also the VI method). In extreme cases, a RHC controller formulated with small $\epsilon > 0$ and insufficiently large $T > 0$ can actually destabilize a passively stable NMP system. To see this, consider the linear system:

$$\begin{bmatrix} \dot{\xi} \\ \dot{\eta} \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ -10 & 1 \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u, \quad y = \xi. \tag{4.31}$$

Although the un-driven dynamics are exponentially stable, Lemma 10 predicts that for any $T \geq \Delta t > 0$ the RHC controller $u^{\epsilon}_{T,\Delta t}$ will destabilize the system if $\epsilon > 0$ is too small.

While Lemma 9 deals with 'small' values of $\epsilon > 0$, the following result gives conditions under which 'large' values of $\epsilon > 0$. For passively stable systems such as (4.31), when $\epsilon > 0$ is large the RHC controller will not exert enough control effort to destabilize the system [53] for any value of $T > 0$. However, when the dynamics are passively unstable, 'large' values of $\epsilon > 0$ prevent the RHC controller from exerting enough control effort to stabilize the system unless $T > 0$ is sufficiently large, as the following result demonstrates. In the statement of the following Lemma we will concatinate the original normal form coordinates as $\bar{x} = (\xi, \eta)$ and conscicely represent the dynamics in these coordniates with:

$$\dot{\bar{x}} = \bar{F}(\bar{x}) + \bar{G}(\bar{x})u. \tag{4.32}$$

Lemma 10. *Let Assumptions 14-19 hold. Further assume that the dynamics $\dot{\bar{x}} = -\bar{F}(\bar{x})$ in the normal coordinates are exponentially stable. Then for each $\bar{\epsilon} > 0$ there exists $\bar{T} > 0$ such that for each $\epsilon > \bar{\epsilon}$ and $\bar{T} \geq T \geq \Delta t > 0$ the receding horizon controller $u^{\epsilon}_{T,\Delta t}$ fails to stabilize* (4.16)*.*

We combine the preceding results to obtain the following:

Theorem 7. *Let Assumptions 14-19 hold. Further assume the additional hypotheses of Lemmas 9 and 10 hold. Then there exists $\bar{T} > 0$ such that for each $\epsilon > 0$ and $\bar{T} \geq T \geq \Delta t > 0$ the receding horizon controller $u^{\epsilon}_{T,\Delta t}$ fails to stabilize* (4.16)*.*

Thus, unlike in the MP case, NMP dynamics can impose a structural obstacle limiting how small the system designer can make the prediction horizon while ensuring the stability of the closed-loop system. Taken together, the preceding results demonstrate that the presence of NMP dynamics (with respect to the outputs chosen when synthesizing the cost function) limits the capabilities of the designer and restricts the set of design trade-offs that can be exploited. This roughly matches the chief qualitative separation between MP and NMP systems highlighted by the cheap control literature, but leads to a philosophically different interpretation. In the cheap control literature, the existence of performance limitations for NMP systems is taken directly as a measure of how fundamentally difficult it is to drive

both the outputs and zeros to the origin. Our results further this perspective. Not only are there fundamental limitations to what is achievable in closed-loop, but these limitations also apply to the synthesis of stabilizing controllers which rely on computation to develop a strategy which attempts to stabilizing both the output and zero subsystems.

Relaxing Assumptions

Finally we briefly discuss when the technical assumptions made in the chapter can be relaxed and when there are obstacles to doing so. First, let us discuss the Assumption 16, which stipulates that the system can be put into strict feedback form. For more general nonlinear systems of the form (4.7), driving the outputs to zero with high-gain feedback control may destabilize the zero dynamics, even when the system is exponentially MP, due to the well-documented *peaking phenomena* (see.[96] for a comprehensive discussion). Thus, without additional structural assumptions about the interconnection between the two subsystems we cannot guarantee that the system does not suffer from performance limitations.

Next, consider Assumption 14, which stipulates that the number of inputs equals the number of outputs. As long as our other structural assumptions hold, there is little difficulty in extending our results to the case where there are fewer inputs than outputs, so long as the outputs can be decoupled by state feedback. On the other hand, when there are more outputs than inputs the results from [16] indicate that the input-output system will suffer from performance limitations, as the output channels cannot be decoupled by state feedback and driven to zero at arbitrary rates.

Assumption 15, which stipulates that each of the outputs has the same relative degree, is made primarily to streamline analysis. When the outputs have different relative degrees, instead of inducing a fast-slow system of the form (4.14), the cheap control problem induces a singular perturbation problem with multiple time-scale which is much more cumbersome to analyze [89]. Nonetheless, we believe an extension of our results to these cases is possible.

Finally, we discuss the tacit Assumption that has been made throughout the chapter which assume that there are no constraints on the inputs or states. When either the state or the input is constrained performance limitations may arise, as there may be a limit to how quickly the outputs can be driven to zero [16]. Although we leave a more detailed characterization of these scenarios to future work, the perspective taken in this chapter forms the basis for explaining why constraints can make it fundamentally difficult, from a computational perspective, to obtain a stabilizing controller. The experiments with modern reinforcement learning methods detailed below corroborate this perspective empirically.

Value Iteration and Other Computational Considerations

In practice, numerical VI algorithms typically use a very small time-step $\Delta t > 0$. Using the correspondence between RHC and VI discussed in Section 4.1, in the MP case Theorem 5 indicates that for a fixed Δt we can reduce the number of iterations VI requires to produce

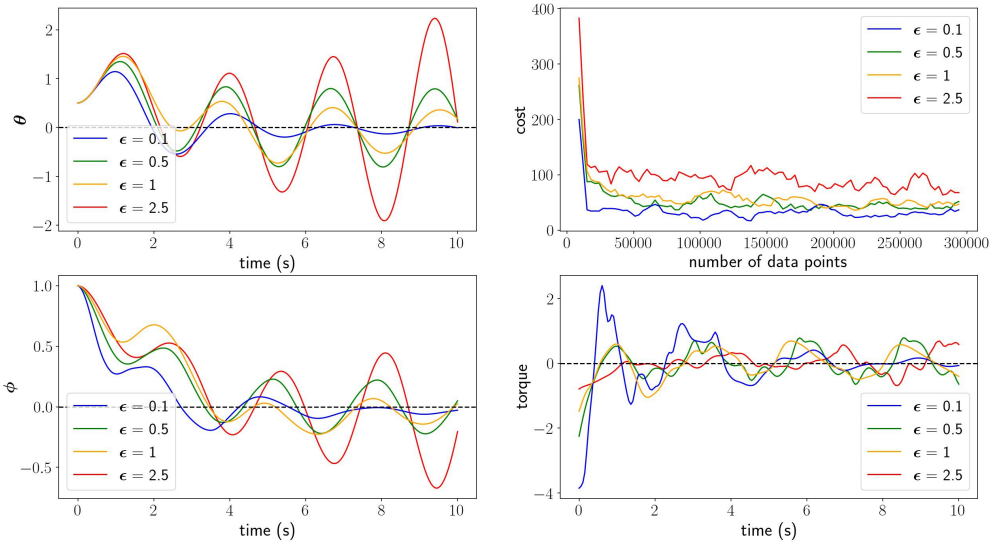

Figure 4.1: Flexible link manipulator without friction when $y = x_1$.

a stabilize controller by decreasing $\epsilon > 0$. In particular, the result demonstrates that there exists $\bar{\epsilon} > 0$ sufficiently small such that for each $\epsilon \in (0, \bar{\epsilon}]$ VI will produce a controller which stabilizes the system after only one iteration, for any fixed Δt. On the other hand, Theorem 7 indicates that there may be a lower-bound to how many iterations are required to stabilize the system in the NMP case.

An important direction for future work is to characterize how issues related to numerical discretization affect the qualitative results developed here, where we have studied idealized versions of RHC and VI in which continuous-time optimal control problems are solved as a subroutine. While this has allowed us to clearly characterize when interactions between the cost function and the feedback geometry of the system lead to certain fundamental limitations, the high-gain feedback controllers produced by VI and RHC as we take $\epsilon \to 0$ will lead to numerical stability issues for practical implementations of these methods. For example, in the face of stiff dynamics grid-based VI methods require a very fine mesh to maintain numerical stability, which increases the computational burden of the method. Thus, broadly speaking, we should expect the limitations of numerical approximations schemes to add an additional layer of computational bottlenecks to the ones considered here. The results presented here are best interpretted as a lower-bound for nonlinear optimal control which stems solely from the geometry of the underlying differential equations.

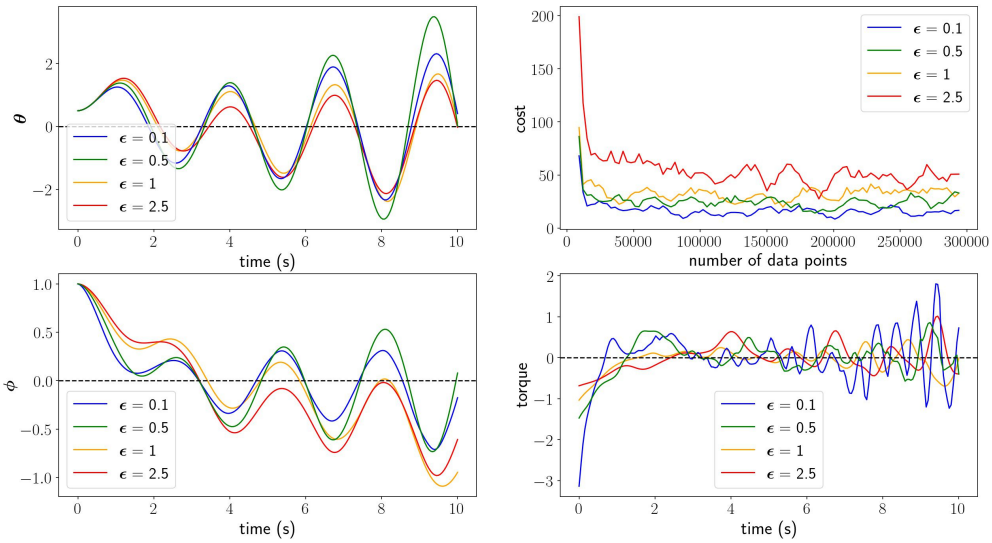

Figure 4.2: Flexible link manipulator without friction when $y = x_3$.

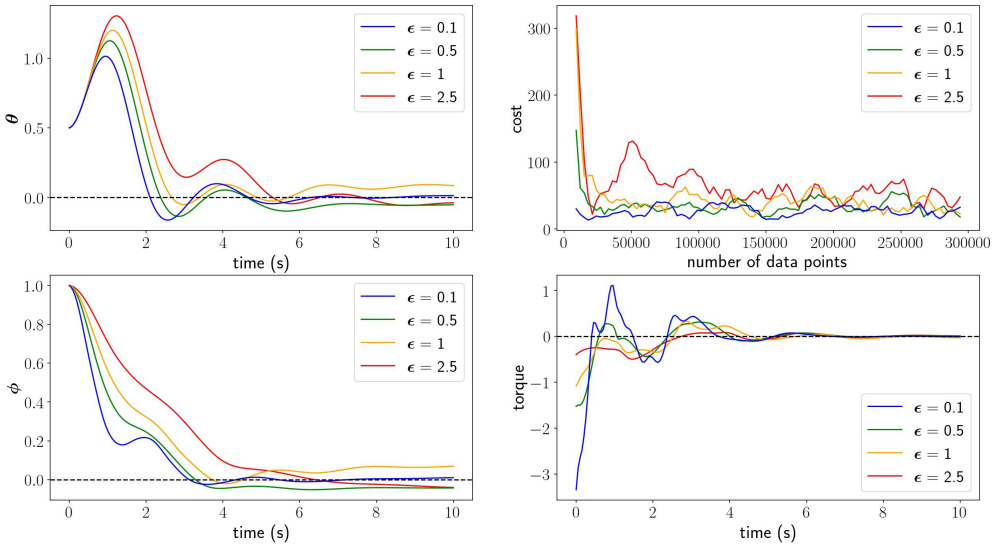

Figure 4.3: Flexible link manipulator with friction with $y = x_1$.

4.3 Numerical Experiments with Reinforcement Learning

While the preceding theoretical analysis applies only to RHC and VI methods, it is reasonable to conjecture that the trade-offs and fundamental limitations we have identified will appear in one form or another for other methods which seek to approximate infinite hori-

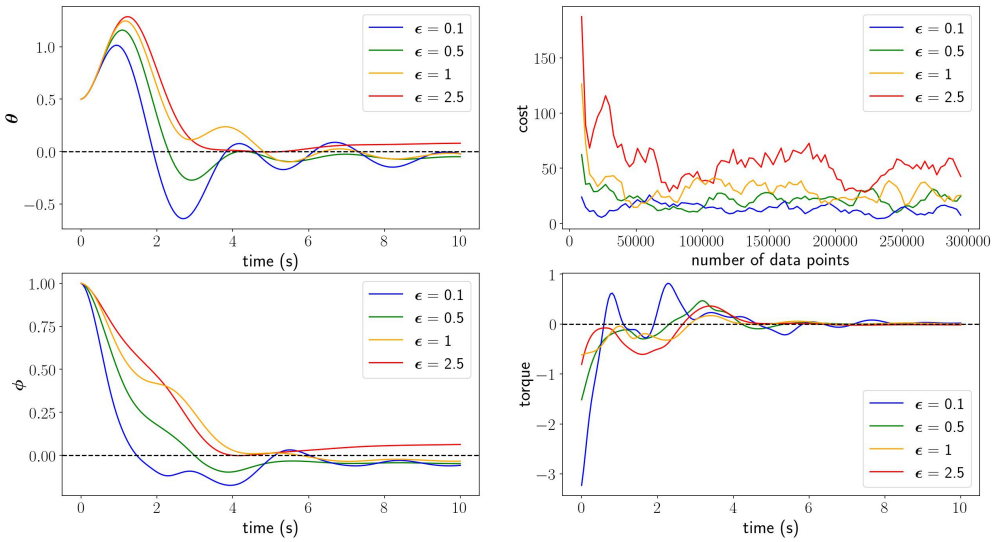

Figure 4.4: Flexible link manipulator with friction with $y = x_3$.

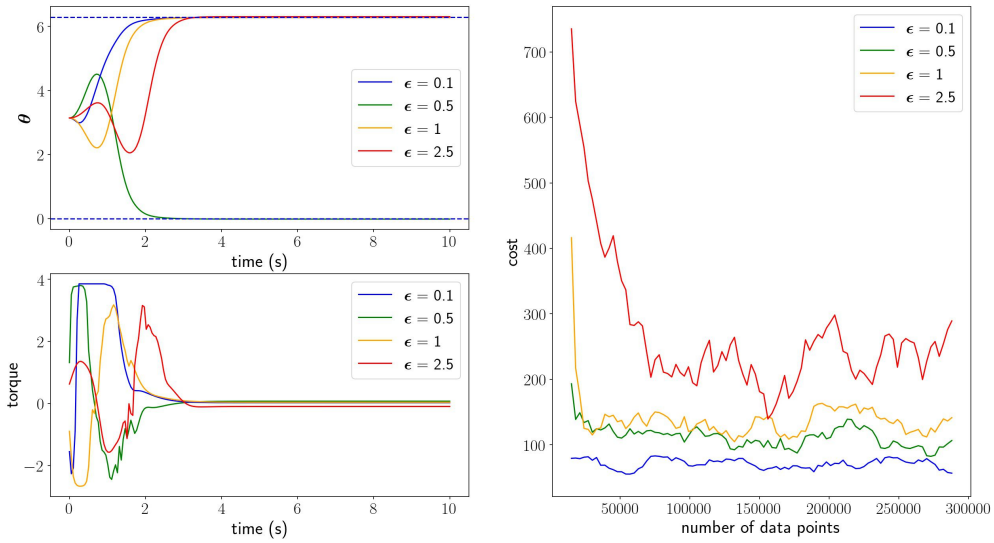

Figure 4.5: Inverted pendulum without input constraints and $y = x_1$.

zon optimal controllers. To test this hypothesis, we now investigate how the choice of cost function impacts the number of samples needed by modern reinforcement learning methods to learn a stabilizing controller. These methods are best viewed as noisy approximations to dynamic programming [14]. Specifically, the following experiments use the soft actor-critic algorithm [37], which can be viewed as an approximation to the policy iteration algorithm [14]. **Inverted Pendulum**: We first consider the dynamics of an inverted pendulum. The states are $(x_1, x_2) = (\theta, \dot{\theta})$, where θ is the angle of the arm from vertical. Units have been

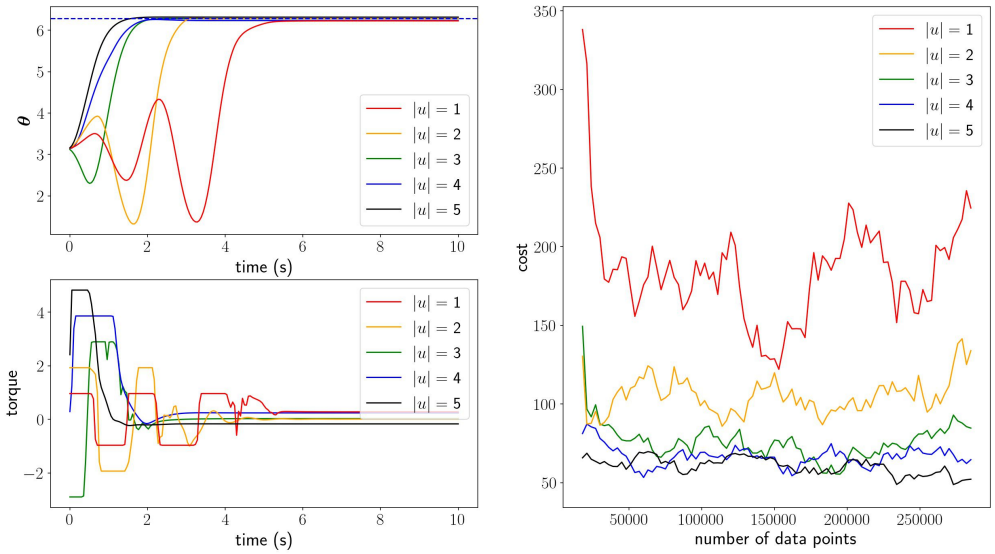

Figure 4.6: Inverted pendulum with input constraints and $y = x_1$.

normalized so that the model is of the form:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \sin(x_2) + u \end{bmatrix},$$

Note that the system is fully-state linearizable with the output $y = x_1$.

Flexible Link Manipulator: We consider a model for a flexible link manipulator which can have both MP and NMP outputs. The state is $(x_1, x_2, x_3, x_4) = (\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2)$, where θ_1 is the angle of the arm from vertical and θ_2 is the internal angle of the motor. The dynamics are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ sin(x_1) + K(x_3 - x_1) - \beta_1 x_2 \\ x_4 \\ K(x_1 - x_3) - \beta_2 x_4 + u \end{bmatrix},$$

where $K > 1$ is a spring coefficient used to model the flexibility of the joint and $\beta_1, \beta_2 \geq 0$ are friction coefficients. One may observe that if the output $y = x_1$ is chosen then the system is full state lineraizable. However when the output $y = x_3$ is chosen the system has a relative degree of two and the zeros are also two dimensional. In this case a Jacobian linearization at the origin reveals that when the model is friction-less ($\beta_1 = \beta_2 = 0$) the system is NMP but when damping is present ($\beta_1, \beta_2 > 0$) the system is MP.

Flexible Link Manipulator Without Friction

We run experiments for the output $y = x_1$ in Figure 4.1 and the output $y = x_3$ in Figure 4.2. In each figure we plot the results obtained by training a controller with the soft actor-critic

algorithm using different values of the weighting parameter $\epsilon > 0$. The upper-right plot in each figure depicts the average cost obtained by the algorithm after obtaining access to different numbers of samples. While the left-most plots depict a trajectory generated by the best-performing controller that was obtained after 300,000 samples of the dynamics. As the figures clearly show, the reinforcement learning algorithm struggles to learn a stabilizing controller when $y = x_3$ for all values of $\epsilon > 0$ that were tested. However when $y = x_1$ the algorithm is able to rapidly learn a stabilizing controller for small values of $\epsilon > 0$ but again struggles when the parameter is large. Thus, we conclude that the flat output $y = x_1$ is a 'better' choice of output, and observe that in these preliminary investigations the trade-offs and limitations we characterized above appear to hold when reinforcement learning algorithms are employed.

Flexible Link Manipulator With Friction

Next we consider the flexible link manipulator with friction with $y = x_1$ in Figure 4.3 and $y = x_3$ in Figure 4.4. In both cases the algorithm is able reliably learn a stabilizing controller as the dynamics are minimumphase. We again observe that the convergence of the learning algorithm is generally faster for small values of ϵ. When compared to the previous experiments, we observe that the added passivity from the friction terms generally makes it easier to learn stabilizing controllers for the system, regardless of the output that is chosen since both choices now yield minimumphase behavior.

Inverted Pendulum With and Without Input Constraints

Next we consider the inverted pendulum without input constraints in Figure 4.5 and with input constraints in Figure 4.6 (where the different colors correspond to different input bounds of the form $|u| \leq k$). In both cases we choose $y_1 = x_1$. For the unconstrained case we see that as ϵ decreases the algorithm is able to rapidly learn a stabilizing controller. For the constrained case, where $\epsilon = 0.1$, we see that as the input constraints are decreased the algorithm takes longer to learn a stabilizing controller. Interestingly, we observed that as we increased ϵ and decreased the bounds on the inputs, the learned controllers display 'swing-up' behavior where the arm pumps multiple times before swinging up.

4.4 Future Work

Ultimately, the long-term goal for this line of work is to provide the foundation for a geometric complexity for nonlinear optimal control, using the notion of complexity in foundational works such as [21]. At the end of the day, an exponential dependence on the input and state dimensionalities will be unavoidable for nonlinear optimal control methods such as reinforcement learning. This is because these methods must 'fill' the state and input space to obtain a nearly optimal controller. However, the perspective developed in this chapter has

the potential to illuminate to what extent design choices made by the user can ameliorate these challenges by designing objectives which can be optimized more greedily to stabilize the system. Even if there are unavoidable exponential dependencies, there existence makes the need to leverage available design decisions to the utmost even more pressing.

4.5 Missing Proofs

Proof of Theorem 4

Let u be such that $J_T(x_0, u) = V_T(x_0)$. Denote $\phi(\tau) = \phi(\tau, x_0, u)$ as the evolution of $x(t)$ for τ time units under control input u starting from x_0. Consider $j \in [0, T - \Delta t]$. Then

$$\begin{aligned}
&V_T(x(\Delta t)) - V_T(x(0)) = V(\phi(\Delta t)) - \int_0^T l(\tau) \cdot d\tau \\
&\leq -\int_0^T l(\tau) \cdot dt + \int_{\Delta t}^{T-j} l(\tau) \cdot d\tau \\
&+ \min_{\tilde{u}} \int_{T-j}^{T+\Delta t} l(\phi(\tau, \phi(T-j), \tilde{u}), \tilde{u}(\tau)) \cdot d\tau \\
&\leq -\int_0^{\Delta t} l(\tau) \cdot d\tau + V_j(\phi(T-j)) \\
&\leq \bar{\alpha}(\sigma(\phi(T-j))) - \int_0^{\Delta t} l(\tau) \cdot d\tau
\end{aligned}$$

As $V_T(\phi(0)) \geq \int_0^T l(t)dt$:

$$\begin{aligned}
W(\phi(T)) - W(\phi(0)) &\leq -k \int_0^T \|x(t)\|_2^2 dt + \int_0^T l(t)dt \\
&= V_T(\phi(0)) - k \int_0^T \|x(t)\|_2^2 dt
\end{aligned}$$

Noting that $0 \leq W(x) \leq \bar{\alpha}_W(\sigma(x))$ and $V_T(x) \leq \bar{\alpha}(\sigma(x))$, we can thus rearrange and bound terms to show the following:

$$k \int_0^T \|x(t)\|_2^2 dt \leq (\bar{\alpha}_W + \bar{\alpha}) \circ \sigma(\phi(0))$$

Now consider $t^* \in [0, T]$ such that

$$t^* = \arg \min_{t \in [0,T]} \|x(t)\|_2^2$$

which exists by continuity of $x(t)$. Then we can note that

$$\|x(t^*)\|_2^2 \leq \frac{(\bar{\alpha}_W + \bar{\alpha}) \circ \sigma(\phi(0))}{kT}$$

Taking $j = T - t^*$ we have that

$$\|x(T-j)\|_2^2 \leq \frac{(\bar{\alpha}_W + \bar{\alpha}) \circ \sigma(\phi(0))}{kT}$$

We can combine this with the previous result on $V_T(x(\Delta t)) - V_T(x(0))$ to get the following:

$$\begin{aligned} V_T(x(\Delta t)) - V_T(x(0)) \leq &- \int_0^{\Delta t} l(\tau) \cdot d\tau \\ &+ \bar{\alpha}(\frac{(\bar{\alpha}_W + \bar{\alpha}) \circ \sigma(\phi(0))}{kT}) \end{aligned}$$

where we leverage the fact that $\bar{\alpha}$ is non-decreasing.

Now note that

$$V_T(x(0)) = \int_0^{\Delta t} l(t)dt + V_{T-\Delta t}(x(\Delta t))$$

Hence $\exists \bar{T} \geq 0$ s.t $\forall T \geq \bar{T}$

$$\begin{aligned} V_T(x(\Delta t)) - V_{T-\Delta t}(x(\Delta t)) &= V_T(x(\Delta t)) - V_T(x_0) \\ &+ \int_0^{\Delta t} l(\tau)d\tau \\ &\leq \bar{\alpha}(\frac{(\bar{\alpha}_W + \bar{\alpha}) \circ \sigma(\phi(0))}{kT}) \end{aligned}$$

Also note that by assumption there exists k_1, k_2 s.t

$$\begin{aligned} W(x) + V_{T-\Delta t}(x) &\leq W(x) + V_\infty(x) \\ &\leq (k_1 + k_2)\|x\|^2 \end{aligned}$$

Hence we can show the following

$$\begin{aligned} &\frac{d}{dt}(W(x) + V_{T-t}(x)) \leq -k\|x\|^2 \\ \implies &\frac{d}{dt}(W(x) + V_{T-t}(x)) \leq -\bar{k}(W(x) + V_{T-t}(x)) \\ \implies &W(x) + V_{T-t}(x) \leq e^{-\bar{k}t}(W(x(0)) + V_T(x(0))) \end{aligned}$$

where $\bar{k} = \frac{k}{k_1+k_2}$. Thus we have the following: Defining $Y = W + V_T$ and $\alpha = (\bar{\alpha}_w + \bar{\alpha})$ completes the proof. Then, use the bound $\underline{\alpha}_W \cdot \sigma(x) \leq W(x) \leq Y_T(x)$ we have

$$Y_T(\Phi(\Delta t)) \leq \left(e^{-\bar{k}\Delta t} + \bar{\alpha}(\frac{(\bar{\alpha}_W + \bar{\alpha})}{\underline{\alpha}_W kT})\right) Y_T(\phi(0)) \tag{4.33}$$

4.6 Performance Bounds

The following performance bounds are crucial for the proofs of our main results, and are state in the $(\tilde{\xi}, \eta)$ coordinates for the state and the $\tilde{u}$ coordinates for the input.

Proof of Lemma 6

We restate the Lemma for convenience. We recall that the system is trivially exponentially stable if it is full-state linearizable, in which case the η terms in the statement of the result and proof can simply be ignored.

Lemma 6. *Let Assumptions 14-19 hold. Further assume that* (4.9) *is exponentially minimum-phase (including full-state linearizable). Then there exist* $\hat{K} > 0$ *such that for each* $0 < \tilde{\epsilon} \leq 1$ *we have for each* $(\tilde{\xi}, \eta) \in \mathbb{R}^n$:

$$\tilde{V}_{\infty}^{\tilde{\epsilon}}(\tilde{\xi}, \eta) \leq \hat{K}\big(\tilde{\epsilon}\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}^{2r}\|\eta\|_2^2\big). \tag{4.22}$$

Proof. To provide the performance guarantee we will apply a sub-optimal feedback linearizaing controller of the the form:

$$\tilde{u} = \tilde{A}^{-1}(\tilde{\xi}, \eta)[-\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}, \eta) + K\tilde{\xi}],$$

to the fast-slow representation of the dynamics (4.14), where K is chosen such that for some $M > 0$, $\|\tilde{\xi}(t)\|_2 \leq Me^{-\frac{t}{\epsilon}}\|\tilde{\xi}(0)\|_2$ for all $t \geq 0$. We will prove the result for the case where the system is not full-state linearizable; the proof for the full-state linearizable case follows by simply ignoring terms related to the zeros in the proof. First, we seek to upper-bound the rate of decay of the zero coordinates. Recall that, by our assumption that $\dot{\eta} = f_0(\eta)$ is exponentially stable, by the converse Lyapunov theorem $\exists c_1, c_2, c_3, c_4 > 0$ and $V(\eta)$ a Lyapunov function s.t. $\forall \eta$:

$$\begin{aligned} c_1\|\eta\|_2^2 \leq V(\eta) \leq c_2\|\eta\|_2^2, \\ \frac{dV(\eta)}{d\eta} f_0(\eta) \leq -c_3\|\eta\|_2^2, \\ \|\frac{dV(\eta)}{d\eta}\|_2 \leq c_4\|\eta\|_2 \end{aligned}$$

Now consider the time derivative of $V(\eta)$ along the full system dynamics:

$$\begin{aligned}
\dot{V}(\eta) &= \frac{dV(\eta)}{d\eta}(f_0(\eta) + g_0(\eta)\tilde{\xi}_1) \\
&\leq -c_3\|\eta\|_2^2 + \|\frac{dV(\eta)}{d\eta}\|_2\|g_0(\eta)\|_2\|\tilde{\xi}_1\|_2 \\
&\leq -c_3\|\eta\|_2^2 + c_4L\|\eta\|_2\|\tilde{\xi}\|_2 \\
-&\leq \frac{c_3}{2}\|\eta\|_2^2 + \frac{2c_4^2L^2}{c_3}\|\tilde{\xi}\|_2^2 \\
&\leq -\frac{c_3}{2c_2}V(\eta) + \frac{2c_4^2L}{c_2c_3}\|\tilde{\xi}\|_2^2 \\
&= - \leq \frac{c_3}{2c_2}V(\eta) + \frac{2c_4^2L^2}{c_2c_3}\|\tilde{\xi}\|^2,
\end{aligned}$$

where we have used the AM-GM inequality and the growth conditions in 19 If we choose $\tilde{\epsilon} > 0$ to be small enough so that $\frac{1}{\tilde{\epsilon}} > \frac{1}{2}\cdot\frac{c_3}{2c_2}$, applying the comparison principle and $\|\tilde{\xi}(t)\|_2^2 \leq Me^{-\frac{1}{\tilde{\epsilon}t}}$ yields:

$$\begin{aligned}
V(\eta(t)) &\leq \exp\bigg(-\frac{c_3}{2c_2}t\bigg)V(\eta(0)) + \frac{2c_4^2L^2}{c_2c_3}\int_0^t \exp\bigg(-\frac{c_3}{2c_2}(t-\tau)\bigg)\|\tilde{\xi}(\tau)\|\cdot d\tau \\
&\leq \exp\bigg(-\frac{c_3}{2c_2}t\bigg)\bigg(V(\eta(0)) + \frac{2M^2c_4^2L^2}{c_2c_3}\int_0^t \exp\bigg(\bigg[\frac{c_3}{2c_2} - \frac{1}{\tilde{\epsilon}}\bigg]\tau\bigg)\|\tilde{\xi}(0)\|_2^2 d\tau\bigg) \\
&\leq \exp\bigg(-\frac{c_3}{2c_2}t\bigg)\bigg(V(\eta(0)) + \frac{2M^2c_4^2L^2}{c_2c_3}\int_0^t \exp\bigg(-\frac{1}{2}\frac{c_3}{2c_2}\tau\bigg)\|\tilde{\xi}(0)\|_2^2 d\tau\bigg) \\
&\leq \exp\bigg(-\frac{c_3}{2c_2}t\bigg)\bigg(V(\eta(0)) + \frac{4M^2c_4^2L^2}{c_3^2}\|\tilde{\xi}(0)\|_2^2\bigg)
\end{aligned}$$

Using $c_1\|\eta(t)\| \leq V(\eta(t)) \leq c_2\|\eta(t)\|_2^2$ this implies:

$$\|\eta(t)\|_2^2 \leq \exp\bigg(-\frac{c_3}{2c_2}t\bigg)\bigg(\frac{c_3}{c_2}\|\eta(0)\|_2^2 + \frac{4M^2c_4^2L^2}{c_2c_3^2}\|\tilde{\xi}(0)\|_2^2\bigg)$$

Next, we seek an upper bound on $\|\tilde{u}(t)\|_2^2$. Recall from Assumption 19 that i) for each $(\tilde{\xi},\eta) \in \mathbb{R}^n$ we have $\sigma_{min}(\tilde{A}^{-1}(\tilde{\xi},\eta)) \geq \frac{1}{\gamma}$ and ii) in the original (ξ,η) normal coordinates we have $b(\xi,\eta) \leq L(\|\xi\| + \|\eta\|)$ for each $(\xi,\eta) \in \mathbb{R}^n$. Further noting that $\|\tilde{\xi}_1(t)\|_2 \leq \|\tilde{\xi}(t)\|_2$ for all $t \geq 0$, we can bound:

$$\begin{aligned}
\tilde{\epsilon}^{2r}\|\tilde{b}(\tilde{\xi},\eta)\|_2^2 &= \tilde{\epsilon}^{2r}\|b(S^{-1}(\tilde{\epsilon})\tilde{\xi},\eta)\|_2^2 \\
&\leq \frac{3}{2}L\tilde{\epsilon}^{2r}(\|S^{-1}(\tilde{\epsilon})\tilde{\xi}\|_2^2 + \|\eta\|_2^2) \\
&\leq \frac{3}{2}L\tilde{\epsilon}^{2r}(\frac{1}{\tilde{\epsilon}^{2(r-1)}}\|\tilde{\xi}\|_2^2 + \|\eta\|_2^2) \\
&= \frac{3}{2}L(\epsilon^2\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}^{2r}\|\eta\|_2^2),
\end{aligned}$$

where we have used Assumption 19 and $(\|S^{-1}(\bar{\epsilon})\tilde{\xi}\|_2 + \|\eta\|_2)^2 \leq \frac{3}{2}(\|\tilde{\xi}\|_2^2 + \|\eta\|_2^2)$. Thus, we have:

$$\begin{aligned}\|\tilde{u}(t)\|_2^2 &\leq \|\tilde{A}^{-1}(\tilde{\xi}(t),\eta(t))\|_2^2 \cdot \left(\frac{3}{2}L^2(\epsilon^2\|\tilde{\xi}(t)\|_2^2 + \tilde{\epsilon}^{2r}\|\eta(t)\|_2^2) + \|K\|_2^2\|\tilde{\xi}(t)\|_2^2\right).\\ &\leq \frac{1}{\gamma}\left((L^2 + \|K\|)\|\tilde{\xi}(t)\|_2^2 + +\tilde{\epsilon}^{2r}L^2\|\eta(t)\|_2^2\right).\end{aligned}$$

Putting these bounds together gives yields:

$$\begin{aligned}\tilde{V}_\infty^{\tilde{\epsilon}}(x) &\leq \int_0^\infty \|\tilde{\xi}_1(t)\|_2^2 + \|\tilde{u}(t)\|_2^2 dt\\ &\leq \int_0^\infty \|\tilde{\xi}_1(t)\|_2^2 dt + \int_0^\infty \leq \frac{1}{\gamma}\left((L^2 + \|K\|)\|\tilde{\xi}(t)\|_2^2 + +\tilde{\epsilon}^{2r}L^2\|\eta(t)\|_2^2\right)dt\\ &\leq \tilde{\epsilon}\tilde{\xi}(0)P\tilde{\xi}(0) + \int_0^\infty \frac{1}{\gamma}\left((L^2 + \|K\|)\|\tilde{\xi}(t)\|_2^2 + +\tilde{\epsilon}^{2r}L^2\|\eta(t)\|_2^2\right)dt\\ &\leq \tilde{\epsilon}\|P\|\|\tilde{\xi}(0)\|_2^2 + \frac{1}{\gamma}\left(\epsilon M^2(L^2 + \|K\|)\|\tilde{\xi}(0)\|_2^2 + \tilde{\epsilon}^{2r}\frac{8(L)^2M^2c_4^2L^2}{c_2^2c_3^3}\|\eta(0)\|_2^2\right),\end{aligned}$$

which demonstrates the desired result. □

Proof of Lemma 7

We restate the Lemma for convienience:

Lemma 7. *Let Assumptions 14-19 hold. Then for each $\bar{T} > 0$, there exists $\bar{K} > 0$ and $\bar{\epsilon} > 0$ such that for each $\bar{T} \geq T > 0$ and $\epsilon \in (0, \bar{\epsilon}]$ we have for each $(\tilde{\xi}, \eta) \in \mathbb{R}^n$:*

$$\tilde{V}_T^{\tilde{\epsilon}}(\tilde{\xi},\eta) \leq \bar{K}\big(\tilde{\epsilon}\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}^{2r}\|\eta\|_2^2\big). \tag{4.23}$$

Proof. We again apply a control of the form:

$$\tilde{u} = \tilde{A}^{-1}(\tilde{\xi}(t),\eta(t))[-\tilde{b}(\tilde{\xi}(t),\eta(t)) + K\tilde{\xi}(t)]$$

where K is chosen such athat $F+GK$ is Hurwitz. Again let the matrix P solve the Lyapunov equation $(F+GK)^TP + P(F+GK) = -I$. By the construction of K we know that there exists $M > 0$ such that $\tilde{\xi}(t) \leq Me^{-\frac{t}{\epsilon}}\|\tilde{\xi}(t)\|$ for each $t \geq 0$. Next we seek to bound the growth of the zeros under the application of this control law:

$$\begin{aligned}\frac{d}{dt}(\|\eta(t)\|_2^2) &= 2\eta(t)^T[f_0(\eta) + g(\eta)\tilde{\xi}] &(4.34)\\ &\leq 2L\|\eta(t)\|_2^2 + 2L\|\eta(t)\|_2\|\tilde{\xi}(t)\|_2 &(4.35)\\ &\leq 3L\|\eta(t)\|_2^2 + 2L\|\tilde{\xi}(t)\|_2^2 &(4.36)\\ & &(4.37)\end{aligned}$$

, where we have used the AM-GM $a \cdot b \leq \frac{1}{2}(a^2+b^2)$ and the growth conditions in Assumption 19 where we have used Assumption 19 the inequality $a \cdot b \leq \frac{1}{2}(a^2+b^2)$. By the comparison principle the preceding inequality yields:

$$\|\eta(t)\|_2^2 \leq e^{3Lt}\|\eta(0)\| + \int_0^t e^{3L(t-\tau)}\|\tilde{\xi}(\tau)\|_2^2 d\tau \tag{4.38}$$

$$\leq e^{3Lt}\bigg(\|\eta(0)\| + \epsilon T M^2 L \|\tilde{\xi}(0)\|\bigg) \tag{4.39}$$

As was done in the proof of Theorem 6, we can obtain a bound for the linearizing controller of the form:

$$\|\tilde{u}(t)\|_2^2 \leq \frac{1}{\gamma}\bigg((L^2 + \|K\|)\|\tilde{\xi}(t)\|_2^2 + +\tilde{\epsilon}^{2r}L^2\|\eta(t)\|_2^2\bigg) \tag{4.40}$$

Thus, we have:

$$\begin{aligned}
\tilde{V}_T^{\tilde{\epsilon}}(\tilde{\xi}(0),\eta(0)) &\leq \int_0^T \|\tilde{\xi}_1(t)\|_2^2 + \|\tilde{u}(t)\|_2^2 dt \\
&\leq \int_0^T \|\tilde{\xi}_1(t)\|_2^2 dt + \int_0^t \|\tilde{u}(t)\|_2^2 dt \\
&\leq \tilde{\epsilon}\tilde{\xi}(0)P\tilde{\xi}(0) + \int_0^T \frac{1}{\gamma}\bigg((L^2 + \|K\|)\|\tilde{\xi}(t)\|_2^2 + +\tilde{\epsilon}^{2r}\|\eta(t)\|_2^2\bigg)dt \\
&\leq \tilde{\epsilon}\|P\|\|\tilde{\xi}(0)\| + \frac{1}{\gamma}\bigg(\epsilon M^2 T^2 e^{Lt} L(L^2 + \|K\|)\|\tilde{\xi}(0)\|_2^2 + L^2 T e^{Lt}\|\eta(0)\|\bigg).
\end{aligned}$$

This demonstrates the desired result. □

Bounds on Small Time Horizons

The following result upperbounds the cost that can be accumulated in short time horizons. Note that the following bound is independent of the control weighting parameter $\epsilon > 0$. Recall that $\bar{x} = (\xi, \eta)$ succinctly represents the original (unscaled) coordinates in the normal form.

Lemma 11. *Let Assumptions 14-19 hold. Then for each $\bar{T} > 0$ and each $\bar{T} > T$ and $\epsilon > 0$ for each $(\xi, \eta) \in \mathbb{R}^n$ we have:*

$$V_T(\bar{x}_0) \leq e^{L^2 T} T \|\bar{x}(0)\|_2^2 \tag{4.41}$$

Proof. We can upper-bound the growth of the state in the $\bar{x} = (\xi, \eta) \in \mathbb{R}^n$ coordinates by chosing $\tilde{u}(\cdot) = 0$ and noting that in this case we have:

$$\frac{d}{dt}\|\bar{x}(t)\|_2^2 = 2\bar{x}(t)^T \bar{F}(\bar{x}) \tag{4.42}$$

$$\leq L\|\bar{x}(t)\|_2^2 \tag{4.43}$$

where we have used the fact that $\|\bar{F}(\bar{x})\| < L$ from Assumption 19. From the preceding bound and the comparison principle we obtain by the comparison principle:

$$\|\bar{x}(t)\|_2^2 \leq e^{L^2 t}\|\bar{x}(t)\|_2^2 \tag{4.44}$$

We can now upperbound the value $\tilde{V}_T$ as a function of the prediction horizon $T > 0$ using the preceding inequality. In particular, suppose we apply the control $\bar{u}_1(\cdot) \equiv 0$. Using the preceding bound on $\|\bar{x}(t)\|_2^2$ We may bound the value function as follows:

$$\begin{aligned}\tilde{V}_T(\bar{x}(0)) &\leq \tilde{J}_T(\tilde{u}(\cdot), \bar{x}(0)) \\ &\leq \int_0^T e^{L^2 T}\|\bar{x}(0)\| dt \\ &\leq T e^{L^2 T}\|\bar{x}(0)\|_2^2 dt\end{aligned}$$

□

Proof of Theorem 5

To simplify notation, throughot the proof we will let $(\tilde{\xi}(\cdot), \eta(\cdot))$ denote the optimal process associated to solving the finite horizon problem from some fixed initial condition $(\tilde{\xi}_0, \eta)$. We we will first study the properties of this trajectory, and then use the results to bound the decay of the overall receding horizon proccess to the origin.

Our proof will make use of the functions $V_1^\epsilon(\tilde{\xi}) = \tilde{\xi}^T P \tilde{\xi}$ where P is chosen so that $(F + MC)^T P + P(F + LC) = -2I$ where $F + MC$ is Hurwitz, and the function $V_2(\cdot)$ comes from a standard exponential stability converse theorem (see e.g. [90]) for the zero dynamics $\dot{\eta} = f_0(\eta)$. Namely, V_2 satisfies the following:

$$\begin{aligned} c_1\|\eta\|_2^2 \leq V_2(\eta) \leq c_2\|\eta\|_2^2 \\ \frac{d}{d\eta}V(\eta)f_0(\eta) \leq -c_3\|\eta\|_2^2 \\ \|\frac{d}{d\eta}V_2(\eta)\|_2 \leq c_4\|\eta\|_2, \end{aligned}$$

for some positive constant $c_1, c_2, c_3, c_4 > 0$. The storage function for the $\tilde{\xi}$ coordinates satisfies:

$$\begin{aligned}\dot{V}_1^{\tilde{\epsilon}}(\tilde{\xi}) &= \tilde{\xi}^T 2P\big[F + G[\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}, \eta)] + \tilde{A}(\tilde{\xi}, \eta)\tilde{u}\big] \\ &= \tilde{\xi}\bigg((F + NC)P + P(F + NC)\bigg)\tilde{\xi} + 2\tilde{\xi}P[\tilde{\epsilon}^r \tilde{b}(\tilde{\xi}, \eta) + \tilde{A}(\tilde{\xi}, \eta)\tilde{u}] + 2\tilde{\xi}^T PNC\tilde{\xi} \\ &\leq -2\|\tilde{\xi}\|_2^2 + \|P\|_2\|\tilde{\xi}\|C\big(\tilde{\epsilon}(\|\tilde{\xi}\|_2 + \|\eta\|) + \|u\|\big) + \|M\|\|P\|\|\tilde{\xi}_1\|_2^2 \\ &\leq -(2 - \tilde{\epsilon}\frac{3}{2}L\|P\| - \frac{1}{2})\|\tilde{\xi}\|_2^2 + \tilde{\epsilon}\frac{1}{2}L\|P\|\|\eta\|_2^2 + \frac{1}{2}L^2\|P\|^2\|u\|_2^2 + \|N\|\|P\|\|\tilde{\xi}_1\|_2^2,\end{aligned}$$

here $L > 0$ is as in Assumption 19 and in the last step we have made repeated use of the inequality $a \cdot b < (a^2 + b^2)/2$. Henceforth choosing $\tilde{\epsilon}$ to be small enough so that $\tilde{\epsilon}\frac{3}{2}\|P\| < \frac{1}{2}$, the preceding inequality can be reduced to

$$\dot{V}^{\epsilon}(\tilde{\xi}) \leq -\|\xi\|_2^2 + \tilde{C}_1\big(\|\tilde{\xi}_1\|_2^2 + \epsilon\|\eta\|_2^2 + \|u\|_2^2\big) \tag{4.45}$$

for some $\tilde{C}_1 > 1$ sufficiently large.

Now, consider the composite function:

$$\mathcal{V}^{\epsilon}(\tilde{\xi}, \eta, t) = \frac{1}{\tilde{C}_1} V_1^{\epsilon}(\tilde{\xi}) + V_{T-t}^{\epsilon}(\tilde{\xi}, \eta), \tag{4.46}$$

whose time derivative is bounded by:

$$\dot{\mathcal{V}}(\tilde{\xi}, \eta, t) \leq -\frac{1}{\tilde{C}_1}\|\xi\| + \epsilon\|\eta\| \tag{4.47}$$

$$\leq -\frac{1}{\tilde{C}_1}(\|\xi\| + \epsilon\|\eta\|) + 2\epsilon\|\eta\|_2^2. \tag{4.48}$$

Next, note that

$$\tilde{c}_1\epsilon\|\tilde{\xi}\|_2^2 \leq \mathcal{V}(\tilde{\xi}) \leq \tilde{c}_2(\epsilon\|\tilde{\xi}\|_2^2 + \epsilon^2\|\eta\|_2^2) \tag{4.49}$$

where

$$\tilde{c}_1 = \frac{1}{\tilde{C}_1}\lambda_{min}(P), \quad \tilde{c}_2 = \hat{K} + \frac{1}{\tilde{C}_1}\lambda_{max}(P), \tag{4.50}$$

where the constant $K_1 > 0$ is from the bound on V_{∞}^{ϵ} in Lemma 6. Thus, we have:

$$\dot{\mathcal{V}}(\tilde{\xi}, \eta, t) = -\frac{1}{\epsilon\tilde{c}_1\tilde{C}_1}\mathcal{V}(\tilde{\xi}, \eta, t) + 2\epsilon\|\eta\|_2^2, \tag{4.51}$$

which, by the comparison principle, yields:

$$\mathcal{V}(\tilde{\xi}, \eta, t) \leq e^{-\frac{1}{\epsilon\tilde{c}_2\tilde{C}_1}t}\mathcal{V}(\tilde{\xi}(0), \eta(0), 0) + 2\epsilon\int_0^t e^{-\frac{1}{\tilde{\epsilon}\tilde{c}_2\tilde{C}_1}t}\|\eta(t)\|_2^2 dt. \tag{4.52}$$

Next let us consider:

$$\dot{V}_2(\eta) = \frac{d}{d\eta}V_2(\eta)[f_0(\eta) + g_0(\eta)\tilde{\xi}_1] \tag{4.53}$$

$$\leq -c_3\|\eta\|_2^2 + Lc_4\|\eta\|_2^2\|\tilde{\xi}_1\|_2^2 \tag{4.54}$$

$$\leq -(c_3 - \frac{1}{2})\|\eta\|_2^2 + L^2c_4^2\|\tilde{\xi}_1\|_2^2 \tag{4.55}$$

$$\leq -\frac{1}{c_2}V_2(\eta) + L^2c_4^2\|\tilde{\xi}_1\|_2^2 \tag{4.56}$$

$$\leq -\tilde{C}_2V_2(\eta) + \tilde{C}_3\|\tilde{\xi}_1\|_2. \tag{4.57}$$

where $L > 0$ is as in Assumption 19, and we have used in the second inequality $a \cdot b \leq (a^2 + b^2)/2$, and in the final inequality we have used $c_3 > 2$ and chosen $\tilde{C}_2 > 0$ to be a sufficiently small constant, and $\tilde{C}_3 > 0$ to be sufficiently large. By applying the comparison principle and integrating the preceding inequality we can obtain:

$$V_2(\eta(t)) \leq e^{-\frac{1}{\tilde{C}_2}t} V_2(\eta(0)) + \tilde{C}_3 \int_0^t e^{-\frac{1}{\tilde{C}_2}t} \|\tilde{\xi}(t)\|_2^2 dt \tag{4.58}$$

$$\leq e^{-\frac{1}{\tilde{C}_2}t} V_2(\eta(0)) + \epsilon \tilde{C}_3 \hat{K} \big(\|\tilde{\xi}(0)\|_2^2 + \|\eta(0)\|_2^2\big), \tag{4.59}$$

$$= \big(e^{-\frac{1}{c_3}t} + \frac{\epsilon \tilde{C}_3 \hat{K}}{c_1}\big) V_2(\eta(0)) + \epsilon \tilde{C}_3 \hat{K} \|\xi(0)\|_2^2 \tag{4.60}$$

where the constant $\hat{K} > 0$ is as in Lemma 6 bounding the growth of the infinite horizon value function, and we have used the fact that $\int_0^T \|\tilde{\xi}(t)\| dt < \tilde{J}_T^\epsilon(\tilde{\xi}(0), \eta(0)) < \tilde{V}_\infty^\epsilon(\tilde{\xi}(0), \eta(0))$.

Note that the preceding equation demonstrates that for each $t \in [0, \Delta t]$:

$$\|\eta(t)\|_2^2 \leq \frac{3}{2c_1} \left(\|\tilde{\xi}(0)\|_2^2 + \|\eta(0)\|_2^2\right) \tag{4.61}$$

, if we choose ϵ to be small enough so that $\epsilon \tilde{C}_3 \hat{K} < \frac{1}{2}$ henceforth.

Combining the preceding inequality, the preceding bound with (4.52) gives us:

$$\mathcal{V}(\tilde{\xi}, \eta, t) \leq e^{-\frac{1}{2\tilde{c}_1 \tilde{C}_1 \epsilon}t} \mathcal{V}(\tilde{\xi}, \eta, 0) \tag{4.62}$$

$$+ \epsilon \frac{3}{c_1} \left(\|\tilde{\xi}(0)\|_2^2 + \|\eta(0)\|_2^2\right) \int_0^t e^{-\frac{1}{\tilde{C}_1 \epsilon \tilde{c}_2}t} dt \tag{4.63}$$

$$\leq e^{-\frac{\alpha}{2\tilde{c}_2 \epsilon}t} \mathcal{V}(\tilde{\xi}, \eta, 0) + \frac{3\epsilon^2}{2\alpha c_2} \left(\|\tilde{\xi}(0)\|_2^2 + \|\eta(0)\|_2^2\right). \tag{4.64}$$

Next, if we choose ϵ to be small enough so that $-\epsilon \frac{2\tilde{c}_2}{\alpha} \ln \epsilon < \Delta t$ then from the preceding bound and (4.49) we can obtain:

$$\|\xi(\Delta t)\|_2^2 \leq \epsilon \tilde{C}_4 \left(\|\tilde{\xi}(0)\|_2^2 + \|\eta(0)\|_2^2\right), \tag{4.65}$$

where $\tilde{C}_4 > 0$ is chosen to be sufficiently large. We can then transform this into abound of the form:

$$V_1^\epsilon(\tilde{\xi}(\Delta t)) \leq \epsilon \tilde{C}_5 \left(V_1^\epsilon(\tilde{\xi}(0)) + V_2(\eta(0))\right) \tag{4.66}$$

where $\tilde{C}_5 > 0$ is once again a sufficiently large constant.

Moreover, if we choose ϵ to be small enough so that $\frac{\epsilon \tilde{C}_1 K_1}{c_1} < 1 - e^{-\frac{1}{c_3}\Delta t}$ then using equation (4.60) and the bounds on V_2 in (4.58) we can bound:

$$V_2(\eta(\Delta t)) \leq \rho V_2(\eta(0)) + \tilde{C}_6 V_1^\epsilon(\tilde{\xi}(0)) \tag{4.67}$$

for some $\tilde{C}_6 > 0$ sufficiently large and $0 < \rho < 1$.

Next, let $(\tilde{\xi}_k, \eta_k)$ denote the k-th iterates of the receding horizon process. By telescoping the bounds in (4.67) and (4.66) we observe that for each k we will have $V_1^{\tilde{\epsilon}}(\tilde{\xi}_k) \leq \hat{x}_k^2$ and $V_2(\eta_k) \leq \hat{x}_k^2$ where $\hat{x}_k^1$ and $\hat{x}_k^2$ are the positive solutions to the following linear discrete time system:

$$\begin{bmatrix} \hat{x}_{k+1}^2 \\ \hat{x}_{k+1}^2 \end{bmatrix} = \underbrace{\begin{bmatrix} \tilde{\epsilon}\tilde{C}_5 & \tilde{\epsilon}\tilde{C}_5 \\ \tilde{C}_6 & \rho \end{bmatrix}}_{A(\tilde{\epsilon})} \begin{bmatrix} \hat{x}_k^1 \\ \hat{x}_k^2 \end{bmatrix} \qquad \begin{bmatrix} \hat{x}_0^1 \\ \hat{x}_0^2 \end{bmatrix} = \begin{bmatrix} V_1^{\tilde{\epsilon}}(\tilde{\xi}_0) \\ V_2(\eta_0) \end{bmatrix}. \tag{4.68}$$

The solutions $(\hat{x}_k^2, \hat{x}_k^2)$, and consequently the discrete iterate of the RHC proccess $(\tilde{\xi}_k, \eta_k)$ will tend to zero exponentially if the matrix $A(\tilde{\epsilon})$ has both eigen values strictly inside the unit disk. The eigenvalues of the matrix are given by $\left[(\tilde{\epsilon}\tilde{C}_5 + \rho) \pm \sqrt{(\tilde{\epsilon}\tilde{C}_5 + \rho) - 4\tilde{\epsilon}\tilde{C}6}\right]/2$. By inspection one can easily deduce that the eigenvaules will have a magnitude less than 1 for $\tilde{\epsilon}$ sufficiently small. This concludes the proof.

Proof of Lemma 9.

Let $T \geq \Delta t \geq 0$ be fixe as in the statement of the theorem. Since the zero dynamics $\dot{\eta} = f_0(\eta)$ are exponentially unstable, the time reversed dynamics $\dot{\eta} = -f_0(\eta)$ must be exponentially stable and thus there must exist a Lyapunov function V_2 which satisfies:

$$\begin{aligned} \hat{c}_1\|\eta\|_2^2 \leq V_2(\eta) \leq \hat{c}_2\|\eta\|_2^2 \\ -\frac{d}{d\eta}V(\eta)f_0(\eta) \leq -\hat{c}_3\|\eta\|_2^2 \\ \|\frac{d}{d\eta}V_2(\eta)\|_2 \leq \hat{c}_4\|\eta\|_2, \end{aligned}$$

Calculating the time derivative of the Lyapunov function along the dynamics yields:

$$\begin{aligned} \dot{V}_2(\eta) &= \frac{d}{d\eta}V_2(\eta)[f_0(\eta) + g_0(\eta)\tilde{\xi}] && (4.69) \\ &\geq \hat{c}_3\|\eta\|_2^2 + C\hat{c}_4\|\eta\|_2\|\tilde{\xi}_1\|_2 && (4.70) \\ &\geq \bar{C}_5 V_2(\eta) - C_6\|\tilde{\xi}(t)\|_2^2 && (4.71) \end{aligned}$$

for appropriately chosen constants $\bar{C}_5, \bar{C}_6 > 0$. Applying the comparison principle yields:

$$\begin{aligned} V_2(\eta(\Delta T)) &\geq e^{\bar{C}_5 T}\|V_2(\eta(0))\|_2^2 - \bar{C}_6\int_0^T e^{\bar{C}_5(T-t)}\|\tilde{\xi}(t)\|_2^2 dt && (4.72) \\ &\geq e^{\bar{C}_5 T}V_2(\eta(0)) - \bar{C}_7\left(\tilde{\epsilon}\|\tilde{\xi}(0)\|_2^2 + \tilde{\epsilon}^2\|\eta(0)\|_2^2\right) && (4.73) \end{aligned}$$

where the constant $\bar{C}_7 > 0$ is once again chosen to be sufficiently large and we have used the second upper-bound for the non-minimum-phase case in 7. Thus, there exists a constant $\bar{C}_8 > 1$ sufficiently such that for each $\tilde{\epsilon} > 0$ sufficiently small we have:

$$V_2(\eta(\Delta T)) \geq \bar{C}_8 V_2(\eta(0)) - \bar{C}_7 V_1^{\tilde{\epsilon}}(\tilde{\xi}(0)), \tag{4.74}$$

where $V_1^{\tilde{\epsilon}}$ is as in the proof of Theorem 7 above and of the form $V_1^{\tilde{\epsilon}}(\tilde{\xi}) = \tilde{\xi}^T P \tilde{\xi}$ for an appropriate positive definite matrix P. Note, that if $\bar{C}_7 V_1^{\tilde{\epsilon}}(\tilde{\xi}(0)) \leq \bar{C}_8 V_2(\eta(0))$ then we will have $\hat{V}_2(\eta(T)) > \hat{V}_2(\eta(0))$. This will be our basis for demonstrating that the closed-loop system is unstable for ϵ sufficiently small.

Next, we claim that there exists $\bar{C}_8 > 0$ sufficiently large such that for each $\tilde{\epsilon} > 0$ sufficiently small we have:

$$V_1^{\tilde{\epsilon}}(\tilde{\xi}(T)) \leq \tilde{\epsilon} \bar{C}_9 \big(V_1^{\tilde{\epsilon}}(\tilde{\xi}(0)) + \hat{V}_2(\eta(0)) \big). \tag{4.75}$$

In particular, a bound of this form can be derived by following the steps used to derive the bound (4.66), except the upper-bound on V_T^{ϵ} in 7 is used in place of the upper-bound use for the minimum-phase case. The details are omitted for brevity.

Next, let $\{(\tilde{\xi}_k, \eta_k)\}_{k=0}^{\infty}$ be the sequence of iterates generated at the sampling instances of the RHC scheme with $\tilde{\xi}_0 = 0$ and $\eta_0 > 0$. Note that in this case we have the condition

$$V_2(\eta_1) > \bar{C}_8 \hat{V}_2(\eta^0) \tag{4.76}$$

will hold in this case by (4.74). Furthermore, by (4.75) we have:

$$\bar{C}_7 V_1^{\tilde{\epsilon}}(\tilde{\xi}_1) \leq \epsilon \bar{C}_9 \bar{C}_7 \hat{V}_2(\eta^0) \leq \epsilon \bar{C}_7 \frac{\bar{C}_9}{\bar{C}_8} V(\eta_1). \tag{4.77}$$

More generally, if we assume that

$$V_1^{\epsilon}(\tilde{\xi}^k) < \frac{C_8}{\bar{C}_9} \hat{V}_2(\eta^k) \tag{4.78}$$

then we will have

$$\hat{V}_2(\eta^{k+1}) > \hat{V}_2(\eta^k). \tag{4.79}$$

Moreover, under this hypothesis, for $\epsilon > 0$ sufficiently small we will have

$$V_1^{\epsilon}(\tilde{\xi}^{k+1}) \leq \frac{\bar{C}_8}{\bar{C}_9} \hat{V}_2(\eta^{k+1}). \tag{4.80}$$

Since (4.78) for $k = 0$, it will also hold for all iterates $k > 0$ for ϵ sufficiently small. Thus, for ϵ sufficiently small (4.79) will hold for each k, indicating that the zeros escape to infinity and that the RHC process fails to stabilize the system.

4.7 Proof of Lemma 10

Unlike the previous proofs, we will work with the representation $\bar{x} = (\xi, \eta)$ of the original normal form. To simplify notation, we will simply let $\bar{x}(\cdot)$ and $\bar{u}(\cdot)$ denote the optimal state trajectory and input for the planning problem the chosen initial condition. Since we are interested in small values of $T > 0$ for the purposes of this proof, we will assume throughout that $\hat{T} > T$, where $\hat{T} > 0$ is some constant. Since the natural dynamics $\dot{\bar{x}} = \bar{F}(\bar{x})$ are exponentially unstable by assumption, there must exist a function V and constants $\hat{c}_1, \hat{c}_2, \hat{c}_3, \hat{c}_4$ such that:

$$\begin{aligned} \hat{c}_1\|\eta\|_2^2 \leq V(\eta) \leq \hat{c}_2\|\eta\|_2^2 \\ -\frac{d}{d\eta}V(\eta)f_0(\eta) \leq -\hat{c}_3\|\eta\|_2^2 \\ \|\frac{d}{d\eta}V(\eta)\|_2 \leq \hat{c}_4\|\eta\|_2, \end{aligned}$$

Calculating the time derivative along the dynamics yields:

$$\begin{aligned} \dot{V}(\bar{x}) &\geq \frac{d}{d\bar{x}}V(\bar{x})[\bar{F}(\bar{x}) + \bar{G}(\bar{x})u] && (4.81) \\ &\geq \hat{c}_3\|\bar{x}\|_2^2 - \hat{c}_4 K\|\bar{x}\|\|u\| && (4.82) \\ &\geq \frac{\hat{c}_3}{2}\|\bar{x}\|_2^2 - \frac{2\hat{c}_4 K}{\bar{c}_3}u && (4.83) \\ &\geq \hat{C}_1\|\bar{x}\|_2^2 - \hat{C}_2\|u\|_2^2 && (4.84) \end{aligned}$$

, for some sufficiently large constants $\hat{C}_1, \hat{C}_2 > 0$.

The preceding equation demonstrates that if $\hat{C}_2\|u(t)\|_2^2 < \bar{C}_1\|\bar{x}(t)\|_2^2$ for each $t \in [0, T]$, then $\bar{V}$ will be strictly increasing when $\bar{x} \neq 0$, which will immediately demonstrates that the RHC process does not stabilize the system, as the value of V will be strictly increasing along trajectories generated by the process. Thus, we will demonstrate how to pick $T > 0$ to be small enough, for the given $\bar{\epsilon} > 0$, such that this condition will hold for all $\epsilon > \bar{\epsilon}$.

Towards this end, for each $\epsilon > \bar{\epsilon}$ we can upper-bound:

$$\int_0^T \|u(t)\|_2^2 \leq \frac{1}{\bar{\epsilon}^2}\bar{V}_T^{\epsilon}(\bar{x}(0)) \leq \frac{1}{\bar{\epsilon}^2}Te^{L^2T}\|\bar{x}(0)\|_2^2 \leq \frac{1}{\bar{\epsilon}^2}\hat{T}e^{L^2\hat{T}}\|\bar{x}(0)\|_2^2 \leq \hat{C}_3\|\bar{x}(0)\|_2^2, \quad (4.85)$$

where we have used Lemma9 and chosen the constant $\hat{C}_3 > 0$ to be sufficiently large.

Next, we employ the Minimum Principle, which in this case dictates that the the optimal input will satisfy:

$$u(t) = -\frac{1}{2}\bar{G}(\bar{x}(t))^T p(t), \quad (4.86)$$

where the costate $p\colon [0,T] \to \mathbb{R}^n$ satisfies the terminal boundary value problem:

$$\dot{p}(t) = \underbrace{-\frac{d}{dx}F^*(\bar{x}(t), p(t))^T p(t) - [\xi(t), \eta(t)]}_{M(t,p(t))} \quad p(t) = 0 \tag{4.87}$$

where

$$F^*(\bar{x}(t), p(t)) = \bar{F}(\bar{x}(t)) - \bar{G}(\bar{x}(t))\bar{G}(\bar{x}(t))^T p(t).$$

Henceforth, we will restrict our attention to the following set:

$$\{\bar{x} \in \mathbb{R}^n \colon \|\bar{x}\| \leq \delta\}, \tag{4.88}$$

where the constant $\delta > 0$ is arbitrary, and we will attempt to argue that the storage function V defined above is strictly increasing along optimal trajectories in this set (except those that originate at $\bar{x} = 0$).

We bound the growth of the cos-state as follows:

$$\frac{d}{dt}\|p(t)\|_2^2 = p(t)^T M(t,(t)) \tag{4.89}$$

$$\leq \hat{C}_5 \bigg((\|p(t)\|^2)^2 + \|p(t)\|_2^2 + \|\bar{x}(t)\|_2^2 \bigg), \tag{4.90}$$

where we have used the fact that $\frac{d}{dx}\bar{F}$ will be bounded uniformly on our chosen set, the constant $\bar{C}_5 > 0$ is chosen to be sufficiently large and depends on $\delta > 0$, and we have repeatedly used the AM-GM inequality. Thus, we may upper-bound the growth of the costate using the Ricatti-type equation:

$$\dot{a}(t) = \hat{C}_5 \bigg(a(t)^2 + a(t)^2 + q(t) \bigg), \quad a(0) = 0 \tag{4.91}$$

where $q(t) = \|\bar{x}(t)\|_2^2$ and we have $\|p(t)\|_2^2 \leq a(t)$. In general, the preceding Ricatti-type ode may not exist for all $t \in [0,T]$, but we can henceforth choose $T > 0$ to be small enough so that this is the case for each initial condition within the chosen ball. Moreover, we can further chose $T > 0$ to be small enough so that we have $\|a(t)\| < 1$ for each $t \in [0,T]$. In this case we will have that:

$$\dot{a}(t) \leq 2a(t) + q(t), \tag{4.92}$$

and thus we can apply the comparison principle to yield the following bound:

$$a(t) \leq e^{2T} \int_0^T q(\tau) d\tau. \tag{4.93}$$

Thus, on short time intervals, we can upperbound the growth of the costate (and thus the input via equation (4.86)) if we can upper bound the growth of the state. Towards this end,

by a standard application of the Bellman-Gronwall inequality we can obtain a bound of the form:

$$\|\bar{x}(t)\| \leq e^{LT}\|\bar{x}(0)\| + e^{LT}\int_0^T \|u(t)\|_2 dt. \tag{4.94}$$

Combining this with (4.85) easily yields a bound of the form:

$$\|\bar{x}(t)\|^2 \leq \hat{C}_6\|\bar{x}(0)\|^2 \quad \forall t \in [0,T]\,, \tag{4.95}$$

where we recall that we have chosen $\hat{T} > T$ throughout the proof. Combining the preceding inequality with (4.93), the inequality $\|p(t)\|^2 \leq a(t)$ and the equality (4.86) readily yields a bound of the form:

$$\|u(t)\|_2^2 \leq \hat{C}_7 T\|\bar{x}(0)\|_2^2 \tag{4.96}$$

for some $\hat{C}_7 > 0$ sufficiently large. Thus, if we choose $T > 0$ to be small enough so that $\hat{C}_1 > \hat{C}_2\hat{C}_7 T$ then (4.86) demonstrates that the value of V will be increasing along the optimal trajectory (and thus also the receding horizon process) at every point in our chosen ball (excluding the origin). This demonstrates that the receding horizon process fails to stabilizing the system when the time horizon is sufficiently small.

Chapter 5

On the Stability of Receding Horizon Control: A Geometric Perspective

Throughout Chapters 2 and 4 there was an implicit assumption that an (approximately) global solution to each optimal control problem that was formulated could be found. The assumption is reasonable if 'brute force' methods such as grid-based (i.e. dense sampling) dynamic programming-based approaches [21] are applied. However, practical implementations of reinforcement learning approaches leverage neural network architectures and policy gradient updates, while practical implementations of receding horizon methods use local descent methods to obtain an open loop control. In both cases, the nonlinear dynamics lead to nonvonvexities in the underlying search space, which prelude guarantees of finding a (nearly) optimal solution where needed. This bottleneck is the primary obstacle preventing the development of practical, scalable convergence guarantees for nonlinear optimal control, in both data-driven scenarios and also instances where the dynamics are assumed to be known.

The goal of this chapter is to link perspectives from geometric control and modern optimization theory, demonstrating how such a link provides new insights into the stability of derivative-based nonlinear RHC implementations. Specifically, by studying how the RHC cost functions interact with both the local and global geometry of the control system, we illustrate how the choice of cost function can lead either to provably stable behavior, or to failure modes where practical RHC control schemes get 'stuck' at undesirable stationary points. Here the term 'practical' refers to the derivative-based search methods favored by practitioners, such as gradient descent or I-LQR, which can only be guaranteed to find approximate stationary points of the underlying objective.

Our negative results are related through two counter-examples while our positive results are given in Theorem 1, which provides sufficient conditions which ensure that all (approximate) first-order stationary points of the RHC cost functional correspond to open-loop state trajectories which decay exponentially to the origin. We use this result to provide stability guarantees for nonlinear RHC when the implementation relies on derivative-based descent methods (Theorem 9), provided that the RHC planning horizon is of sufficient (though

modest) length and a standard 'warm-starting' strategy is employed.

Our sufficient conditions for exponential stability informally require (1) the state and input costs are strongly convex, (2) the Jacobian linearization of the dynamics along every trajectory is uniformly stabilizable, (3) the control cost is sufficiently small when compared to the state cost, (4) the first-order expansion of the system dynamics along each trajectory satisfies a local 'matching condition', meaning that the affine term can be cancelled out by an appropriate choice of input, and (5) the nonlinearity in the input is sufficiently small. We note that assumption (1) is natural in practical implementations of RHC, and some (possibly weaker) stabilizability condition as in (2) is clearly necessary to ensure exponential stability. It is unclear whether (5) is necessary, and we leave its study to future work. We note that conditions $(1) - (2)$ and (4) may be satisfied in some coordinate systems for the state but not others. Thus applying our sufficient conditions (or using them to design a 'good' RHC cost functional) may require finding an appropriate coordinate system for the system.

To shed light on conditions (3) and (4), and to establish their necessity, we examine a class of feedback linearizable systems which satisfy conditions (2), (4) and (5) in an appropriate choice of coordinates. For this class of systems conditions (1) and (4) require that the state costs are strongly convex *in the linearizing coordinates*. To demonstrate the necessity of this strong geometric condition, we examine a model for a flexible-joint manipulator which is full-state linearizable. A natural state cost is designed which is convex in the 'original' non-linearizing coordinates (where condition (4) is violated), but analysis reveals that the cost is non-convex in the linearizing coordinates (where condition (4) is satisfied). Thus, the chosen cost function is in some sense 'incompatible' with the geometry of the system as we are forced to pick a coordinate system in which either (1) or (4) are violated. Due to this mismatch, we are able to identify initial conditions from which derivative-based RHC schemes will fail to stabilize the system and get stuck at undesirable stationary points. To address condition (3) we also investigate a model for the simple inverted pendulum, which is in the class of linearizable systems discussed above. We demonstrate that even when the other four conditions are satisfied RHC may again fail to stabilize the system if the penalty on the input is too large.

Unlike prior works [43] which require global optimality for each RHC planning problem, the stability issues for derivative-based RHC implementations discussed above cannot be overcome by simply increasing the prediction horizon. Indeed, our counterexamples show that conditions (3) and (4) are necessary even when arbitrarily long planning horizons are used. The key difference here is that derivative-based planners can converge to overly-myopic sequences of control inputs, even with long planning horizons, due to the myopic nature of the optimization landscape (i.e., the presence of local minima).

In sum, the results of this chapter indicate that the stability of derivative-based nonlinear RHC schemes may be fragile unless the interaction between the geometry of the control system and cost functions are carefully considered. Fortunately, our positive results indicate that concepts from the geometric control literature may provide constructive techniques for designing RHC cost functionals which provably guide local search algorithms towards

stabilizing solutions.

Further Background on RHC For basic background on RHC, we refer the reader to any of a number of comprehensive reviews on RHC (see, e.g., [68, 70]). Simplifying considerably, previous theoretical nonlinear RHC formulations fall into either *constrained* approaches and *unconstrained* approaches. Constrained RHC formulations directly enforce stability by either constraining the terminal predictive state to lie at the origin [67], or using inequality constraints to force the system into a neighborhood containing the origin, and then stabilizing the system using a local controller [69]. The usual critique of these methods [45] is that the satisfaction of the relevant constraints may be overly demanding computationally in an online implementation. In contrast, unconstrained approaches implicitly enforce stability by either using an appropriate CLF as the terminal cost [45] or a sufficiently long prediction horizon [43]. As alluded to above, most of these stability guarantees require that a globally optimal solution can be found for each prediction problem. Several approaches provide stability guarantees using sub-optimal solutions, but generally require that an initial feasible solution is available [95], which may be restrictive in high-performance real-time scenarios, or require the availability of a CLF [83, 45], which implies that the stabilization problem has already been solved. Thus, in this chapter we study unconstrained RHC formulations which use general terminal costs, and aim to provide stability guarantees which only require that a stationary point of each optimization problem can be found. We feel that this accurately reflects the spirit of optimization-based control—to stabilize the system with minimal system-specific knowledge—as well as the practical computational constraints facing practitioners.

5.1 Preliminaries

This chapter studies control systems of the form

$$\dot{x}(t) = F(x(t), u(t)), \tag{5.1}$$

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ the input, and $\dot{x}(t) = \frac{\mathrm{d}}{\mathrm{d}t}x(t)$ denotes time derivatives. We make the following assumptions about the vector field $F\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$:

Assumption 20. *The origin is an equilibrium point of* (5.1)*, namely,* $F(0, 0) = 0$.

Assumption 21. *The vector field* F *is continuously differentiable. Furthermore, there exist constants* $L_F > 0$ *such that for each* $x_1, x_2 \in \mathbb{R}^n$ *and* $u_1, u_2 \in \mathbb{R}^m$ *we have:*

$$\|F(x_1, u_1) - F(x_2, u_2)\|_2 \leq L_F \left(\|x_1 - x_2\|_2 + \|u_1 - u_2\|_2\right).$$

Taken together, these standard assumptions support the global existence and uniqueness of solutions to (5.1) on compact intervals of time [79, Proposition 5.6.5].

The primary object of study in is chapter will be finite horizon cost functionals $J_T(\cdot; x_0) \colon \mathcal{U}_T \to \mathbb{R}$ of the form

$$J_T(\tilde{u}; x_0) = \int_0^T Q(\tilde{x}(\tau)) + R(\tilde{u}(\tau))\mathrm{d}t + V(\tilde{x}(T)) \\ \text{s.t. } \dot{\tilde{x}}(t) = F(\tilde{x}(t), \tilde{u}(t)), \quad \tilde{x}(0) = x_0, \tag{5.2}$$

where $T > 0$ is a finite prediction horizon, $x_0 \in \mathbb{R}^n$ is the initial condition for (5.1), and the space of admissible inputs is given by $\mathcal{U}_T \colon = \mathcal{L}^2([0,T], \mathbb{R}^m) \cap \mathcal{L}^\infty([0,T], \mathbb{R}^m)$. Here, $Q \colon \mathbb{R}^n \to \mathbb{R}$ is the running cost applied to the state, $R \colon \mathbb{R}^m \to \mathbb{R}$ is the running cost applied to the input, and $V \colon \mathbb{R}^n \to \mathbb{R}$ a penalty for the terminal state. For now we assume that each of these maps is continuously differentiable.

Linearizations and (Approximate) Stationary Points

Next, we briefly review a few basic facts from the calculus of variations which are essential for understanding out results. We endow $\mathcal{L}^2([0,T], \mathbb{R}^m)$ with the usual inner product and norm, denoted $\langle \cdot, \cdot \rangle \colon \mathcal{L}^2([0,T], \mathbb{R}^m) \times \mathcal{L}^2([0,T], \mathbb{R}^m) \to \mathbb{R}$ and $\| \cdot \|_2 : \mathcal{L}^2([0,T], \mathbb{R}^m) \to \mathbb{R}$. Under Assumptions 21 and and the assumption that Q and R are continuously differentiable, directional (Fréchet) derivatives of $J_T(\cdot, x_0)$ are guaranteed to exist [79, Theorem 5.6.8] as there is a well-defined gradient at each point in the optimization space. We denote the directional Fréchet derivative of $J_T(\cdot, x_0)$ at the point $\tilde{u} \in \mathcal{U}_T$ in the direction $\delta u \in \mathcal{U}_T$ by $\mathrm{D}J_T(\tilde{u}; x_0; \delta u)$. The gradient $\nabla J_T(\tilde{u}; x_0) \in \mathcal{L}^2([0,T], \mathbb{R}^m)$ is the unique object satisfying, for each $\delta u \in \mathcal{U}_T$,

$$\mathrm{D}J_T(\tilde{u}; x_0; \delta u) = \int_0^T \langle \nabla J_T(\tilde{u}; x_0)(t), \delta u(t) \rangle \mathrm{d}t, \tag{5.3}$$

or more compactly, $\mathrm{D}J_T(\tilde{u}; x_0; \delta u) = \langle \nabla J_T(\tilde{u}; x_0), \delta u \rangle$. The following notions from the optimization literature are crucial for understanding our technical results:

Definition 5. *We say that an input $\tilde{u}$ is a* **first-order stationary point** *(*FOS*) if $\nabla J_T(\tilde{u}; x_0) = 0$. We say that $\tilde{u}$ is an ϵ-*FOS* if $\|\nabla J_T(\tilde{u}; x_0)\|_2 \leq \epsilon$.*

In practice, derivative-based descent algorithms take an infinite number of iterations to converge to exact stationary points, thus our analysis will primarily focus on the approximate stationary points of $J_T(\cdot, x_0)$, as these can be reached in a finite number of iterations.

Finally, we discuss how to calculate the gradient $\nabla J_T(\tilde{u}; x_0)$ using first-order expansions of the system dynamics and cost functions. Let $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ denote the state-input pair of (5.1) defined on the interval $[0,T]$ such that $\tilde{x}(0) = x_0$ and define for each $t \in [0,T]$

$$\tilde{A}(t) = \frac{\partial}{\partial x} F(\tilde{x}(t), \tilde{u}(t)), \quad \tilde{B}(t) = \frac{\partial}{\partial u} F(\tilde{x}(t), \tilde{u}(t)).$$

Definition 6. *Let $(\tilde{x}(\cdot),\tilde{u}(\cdot))$, $\tilde{A}(\cdot)$ and $\tilde{B}(\cdot)$ be defined as above. We refer to the time-varying linear system $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ as the* **Jacobian linearization** *of the vector field F along the trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$.*

The Jacobian linearization $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ can be used to construct a first-order approximations to trajectories near $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ as follows. Let $\delta u \in \mathcal{U}_T$ be a small admissible perturbation to the input and let $(\hat{x}(\cdot), \hat{u}(\cdot))$ satisfy (5.1) and $\hat{u} = \tilde{u} + \delta u$ and $\hat{x}(0) = \tilde{x}(0) = x_0$. Then a first-order approximation to $\hat{x}(\cdot)$ is given by

$$\hat{x}(\cdot) \approx \bar{x}(\cdot) := \tilde{x}(\cdot) + \delta x(\cdot) \tag{5.4}$$

where $\delta x\colon [0,T] \to \mathbb{R}^n$ solves $\dot{\delta x}(t) = \tilde{A}(t)\delta x(t) + \tilde{B}(t)\delta u(t)$ with $\delta x(0) = 0$, and the approximation in (5.4) suppresses higher-order terms involving δu. Regarding gradients as row vectors, by [79, Theorem 5.6.3] we have

$$\nabla J_T(\tilde{u}; x_0)(t) = p(t)\tilde{B}(t) + \nabla R(\tilde{u}(t)) \tag{5.5}$$

where the *co-state* $p\colon [0,T] \to \mathbb{R}^{1\times n}$ satisfies

$$-\dot{p}(t) = p(t)\tilde{A}(t) + \nabla Q(\tilde{x}(t)) \quad p(t) = \nabla V(\tilde{x}(T)). \tag{5.6}$$

Thus, equations (5.5) and (5.6) reveal that the gradient of the objective can be efficiently computed using a 'backwards pass' along the nominal trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ and the linearizations of the vector field and costs along this curve.

Convex Time-Varying Approximations to $J_T(\cdot, x_0)$

Our primary goal throughout the chapter is study the properties of (approximate) stationary points of $J_T(\cdot; x_0)$, and our primary analytical tool will be a family of convex approximations constructed using the Jacobian linearization around particular trajectories of the system. To begin constructing these approximations, observe that the evolution of the estimate $\bar{x}(\cdot)$ in (5.4) is given by

$$\begin{aligned}
\dot{\bar{x}}(t) &= \dot{\tilde{x}}(t) + \delta\dot{x}(t) \\
&= F(\tilde{x}(t), \tilde{u}(t)) + \tilde{A}(t)\delta x(t) + \tilde{B}(t)\delta u(t) \\
&= \tilde{A}(t)\bar{x}(t) + \tilde{B}(t)\bar{u}(t) + \tilde{d}(t), \text{ where} \qquad (5.7)\\
\tilde{d}(t) &:= F(\tilde{x}(t), \tilde{u}(t)) - \tilde{A}(t)\tilde{x}(t) - \tilde{B}(t)\tilde{u}(t). \qquad (5.8)
\end{aligned}$$

We use these time-varying dynamics to approximate $J_T(\cdot; x_0)$ near the point $\tilde{u}$ with the cost functional $J_T^{\mathtt{jac}}(\cdot; x_0, \tilde{u})\colon \mathcal{U}_T \to \mathbb{R}$ defined as follows:

$$\begin{aligned}
J_T^{\mathtt{jac}}(\bar{u}; x_0, \tilde{u}) &= \textstyle\int_0^T Q(\bar{x}(\tau)) + R(\bar{u}(\tau))\mathrm{d}t + V(\bar{x}(T)) \\
\text{s.t.} \quad \dot{\bar{x}}(t) &= \tilde{A}(t)\bar{x}(t) + \tilde{B}(t)\bar{u}(t) + \tilde{d}(t), \quad \bar{x}(0) = x_0.
\end{aligned} \tag{5.9}$$

The following result, which follows from a direct comparison of the formulas for the gradients of $J_T(\cdot; x_0)$ and $J_T^{\mathtt{jac}}(\cdot; x_0, \tilde{u})$ at the point $\tilde{u}$, motives this construction:

Lemma 12. *For any input $\tilde{u}(\cdot) \in \mathcal{U}_T$ we have*

$$J_T(\tilde{u}; x_0) = J_T^{\mathtt{jac}}(\tilde{u}; x_0, \tilde{u}), \quad \textit{and} \quad \nabla J_T(\tilde{u}; x_0) = \nabla J_T^{\mathtt{jac}}(\tilde{u}; x_0, \tilde{u}).$$

5.2 Sufficient Conditions for Exponentially Decaying First-Order Stationary Points

We begin our analysis by providing sufficient conditions which ensure that *all* (approximate) stationary points of $J_T(\cdot; x_0)$ decay exponentially to the origin at a rate that is independent of $T \geq 0$ and $x_0 \in \mathbb{R}^n$. This is a strong condition which will enable us to provide global exponential stability guarantees for RHC schemes which use derivative-based iterative optimization schemes in Section 5.3. After stating the main result and its proof, we draw the connection to feedback linearization and investigate examples which highlight the necessity of some of our stronger assumptions.

Sufficient Conditions for Exponentially Decaying (Approximate) First-Order Stationary Points

We begin by introducing the Assumptions required for the proof of Theorem 8. We emphasize some of these assumptions are highly dependent on the particular set of coordinates chosen for the state, and may be satisfied in certain coordinate systems but not others. Thus, applying our sufficient conditions requires finding a coordinate system in which the following conditions hold. Later we relate these conditions to the coordinate systems that arise when performing feedback linearization, which succinctly capture the underlying geometry of the control system.

Our first assumption is strong convexity and smoothness of the running and terminal cost functions:

Assumption 22. *We assume that $Q(\cdot), R(\cdot), V(\cdot)$ are twice-continuously differentiable functions, with $Q(0) = R(0) = V(0) = 0$, whose Hessians satisfy the pointwise bounds $\alpha_Q I \preceq \nabla^2 Q \preceq \beta_Q I$, $\alpha_R I \preceq \nabla^2 R \preceq \beta_R I$, and $\alpha_V I \preceq \nabla^2 V \preceq \beta_V I$ for constants $0 < \alpha_Q \leq \beta_Q$, $0 < \alpha_R \leq \beta_R$, and $0 \leq \alpha_V \leq \beta_V$.*

Note that when Assumption 22 is satisfied the optimization in (5.9) is strongly convex. Thus, by Lemma 12, $\tilde{u}$ is a stationary point of $J_T(\cdot, x_0)$ if and only if it is the *global minimizer* of $J_T^{\mathtt{jac}}(\cdot; x_0, \tilde{u})$. Due to the convexity of the approximation, it is much easier to study the properties of stationary points of $J_T(\cdot, x_0)$ using $J_T^{\mathtt{jac}}(\cdot; x_0, \tilde{u})$ rather than the original functional. This observation is a key insight in our poof technique. The following result extends the above discussion to approximate stationary points of $J_T(\cdot, x_0)$:

Lemma 13. *(Approximate* FOS*) Suppose Assumption 22 holds, and that $\tilde{u}$ is an ϵ-*FOS *of $J_T(\cdot; x_0)$. Then,*

$$J_T(\tilde{u}; x_0) \leq \min_u J_T^{\mathtt{jac}}(u; x_0, \tilde{u}) + \tfrac{\epsilon^2}{2\alpha_R}.$$

Proof. Assumption 22 implies that $J_T^{\mathtt{jac}}(\bar{u}; x_0, \tilde{u}) - \alpha_R \|\bar{u}\|^2$ is convex, and thus the Polyak-Łojasiewicz inequality holds: $J_T^{\mathtt{jac}}(\bar{u}; x_0, \tilde{u}) \leq \min_u J_T^{\mathtt{jac}}(u; x_0, \tilde{u}) + \|\nabla J_T^{\mathtt{jac}}(\bar{u}; x_0, \tilde{u})\|_2^2 / 2\alpha_R$. Since $\tilde{u}$ is an ϵ-FOS of $J_T(\cdot; x_0)$ and $\nabla J_T(\tilde{u}; x_0) = \nabla J_T^{\mathtt{jac}}(\tilde{u}; x_0, \tilde{u})$, it follows that $J_T^{\mathtt{jac}}(\tilde{u}; x_0, \tilde{u}) \leq \min_u J_T^{\mathtt{jac}}(u; x_0, \tilde{u}) + \epsilon^2 / 2\alpha_R$. Using $J_T(\tilde{u}; x_0) = J_T^{\mathtt{jac}}(\tilde{u}; x_0, \tilde{u})$ concludes. □

Next we place restrictions on the local structure of the control system along each of its trajectories:

Assumption 23. *Along each system trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ the drift term in* (5.8) *satisfies $\tilde{d}(t) \in \mathrm{range}(\tilde{B}(t))$.*

Assumption 24. *There exists $\gamma > 0$ such that for each time horizon $T \geq 0$, $x_0 \in \mathbb{R}^n$ and system trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ of length T with $\tilde{x}(0) = x_0$ we have*

$$\inf_{\hat{u}(\cdot)} \int_0^T \|\hat{x}(t)\|_2^2 + \|\hat{u}(t)\|_2^2 \mathrm{d}t + \|\hat{x}(T)\|_2^2 \leq \gamma \|x_0\|_2^2, \tag{5.10}$$

where $\dot{\hat{x}}(t) = \tilde{A}(t)\hat{x}(t) + \tilde{B}(t)\hat{u}(t)$ and $\hat{x}(s) = x_0$ and $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ is the Jacobian linearization along $(\tilde{x}(\cdot), \tilde{u}(\cdot))$.

In the language of nonlinear control theory, Assumption 23 is known as a *matching condition* [90, Chapter 9.4]. The assumption implies that the drift term $\tilde{d}(t)$ can be 'cancelled out' by choosing the input $\bar{u}(t) = \tilde{B}^\dagger \tilde{d}(t)$. Meanwhile, the parameter $\gamma > 0$ in Assumption 24 measures the difficulty (in terms of a simple $\mathcal{L}^2$ cost) of stabilizing the Jacobian Linearizations along system trajectories.

Roughly speaking, our first three conditions ensure that the state costs are in a certain sense 'compatible' with the local (first-order) geometry of the control system, meaning that at each point in the optimization space they guide local search algorithms to find an input $\tilde{u}$ which drives the corresponding predictive trajectory $\tilde{x}$ towards the origin. Indeed, by Lemma 13, when Assumptions 22, 23 and 24 all hold, at each point $\tilde{u} \in \mathcal{U}_T$ the functional $J_T(\cdot; x_0)$ has the same local structure (up to first-order approximations) as an optimal control problem with convex costs and stabilizable time-varying dynamics, namely, $J^{\mathtt{jac}}(\cdot; x_0, \tilde{u})$. For this local convex approximation it is much clearer to see how the state costs yield state trajectories which decay to the origin. Our second counter-example investigates a situation where there does not exist a coordinate system in which both Assumption 22 and Assumption 23 can be satisfied simultaneous and local search algorithms can produce predictive trajectories which get 'stuck' at undesirable equilibria.

Our last technical condition, which is made Assuming 23 already holds, effectively bounds how costly it is to 'cancel out' the affine drift term $\tilde{d}(t)$ along each trajectory:

Assumption 25. *There exists $L_x, L_u > 0$ such that for each system trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ defined on $[0,T]$ we have $\|\tilde{B}^\dagger(t)\tilde{d}(t)\| \leq L_x\|\tilde{x}(t)\| + L_u\|\tilde{u}(t)\|$ for each $t \in [0,T]$. Moreover, these constants satisfy*

$$L_u^2 \leq \tfrac{\alpha_R}{8\beta_R} \quad \text{and} \quad L_x^2 \leq \tfrac{\alpha_Q}{8\beta_R} \tag{5.11}$$

In particular, Assumption 25 sates that the cost of rejecting $\tilde{d}(t)$ can only grow linearly with $\tilde{x}(t)$ and $\tilde{u}(t)$. The constant L_x can be made arbitrarily large by re-scaling the relative magnitudes of the state and input costs (so that $\alpha_Q \gg \beta_R$). However, since $\alpha_R \leq \beta_R$, the condition implies that L_u can be at most $\frac{1}{\sqrt{8}}$, which effectively limits how nonlinear the control system is with respect to the input.[1] As our first counter example demonstrates, when Assumption 25 is violated local search algorithm may again get 'stuck' at undesirable stationary points. The intuition for this failure mode is that even when Assumptions 23 and 24 are satisfied if $\tilde{d}(t)$ grows too quickly it may appear 'too costly' (from the perspective of optimization algorithms which only have access to first-order information) to reject $\tilde{d}(t)$ and drive the system to the origin.

Under these assumptions we obtain our main result:

Theorem 8. *Suppose Assumptions 20 to 25 hold. Then for each $T \geq 0$, $x_0 \in \mathbb{R}^n$ and every $\tilde{u} \in \mathcal{U}_T$ which is an ϵ-FOS of $J_T(\cdot; x_0)$ the following hold. If $\alpha_V > 0$, then $\forall s \in [0,T]$,*

$$\|\tilde{x}(s)\|^2 \leq \mathcal{C}_0 \cdot (\mathcal{C}_1 e^{-\frac{s}{\mathcal{C}_1}} \cdot \|x_0\|^2 + \mathcal{C}_2\epsilon^2),$$

where $\mathcal{C}_0 = 6L_F + \frac{2L_F\alpha_Q}{\alpha_R} + \frac{2\alpha_Q}{\alpha_V}$, $\mathcal{C}_1 = 4\gamma\max\{\beta_V, \beta_R, \beta_Q\}$ and $\mathcal{C}_2 = \frac{1}{2\alpha_R}(1 + 8\beta_R\max\{\frac{L_x^2}{\alpha_Q}, \frac{L_u^2}{\alpha_R}\})$. More generally, for any $\alpha_V \geq 0$, it holds that for all $s \in [0,T]$ and $\delta > 0$,

$$\|\tilde{x}(s)\|^2 \leq \mathcal{C}_0^\delta \cdot (\mathcal{C}_1^\delta e^{-\frac{s}{\mathcal{C}_1}} \cdot \|x_0\|^2 + \mathcal{C}_2\epsilon^2)$$

where $\mathcal{C}_0^\delta := 6L_F + \frac{2L_F\alpha_Q}{\alpha_R} + \min\{\frac{2}{\delta}, \frac{2\alpha_Q}{\alpha_R}\}$, $\mathcal{C}_1^\delta := e^{\frac{\delta}{\mathcal{C}_1}}\mathcal{C}_1$.

Note that when $\epsilon = 0$ taking the square root of both sides of either bound in the statement of the theorem demonstrates that the stationary point is exponentially decaying. We emphasize that the rate of decay is uniform across all stationary points corresponding to different initial conditions $x_0 \in \mathbb{R}^n$ and prediction horizons $T > 0$. This uniformity is essential for our RHC stability results in Section 5.3. There we stipulate how long the prediction horizon $T > 0$ and how small the optimality parameter $\epsilon > 0$ must be each time a planning problem is solved to ensure stability.

[1] We remark that the factors of $\frac{1}{8}$ in (5.11) can be replaced by any constant in the interval $(0,1)$ and the proof of Theorem 8 will go through with minor modifications. However fixing a specific constant simplifies the statement of the main result.

Proof of Theorem 8

Let $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ be as in the statement of the theorem, and let $\tilde{\mathcal{V}}(s) := J_{T-s}(\tilde{u}_{[s,T]}; \tilde{x}(s))$ for $s \in [0, T]$. Then the following bounds hold (under the assumptions of Theorem 8):

Lemma 14. *If $\alpha_V > 0$ then for each $0 \leq s' \leq s \leq T$ we have $\|\tilde{x}(s)\|^2 \leq \frac{1}{\alpha_Q}\mathcal{C}_0 \cdot \tilde{\mathcal{V}}(s')$. Alternatively, if $\alpha_V \geq 0$ then for each $0 \leq \delta \leq T$, $s' \leq T - \delta$ and $s' \leq s \leq T$ we have $\|\tilde{x}(s)\|^2 \leq \frac{1}{\alpha_Q}\mathcal{C}_0^\delta \cdot \tilde{\mathcal{V}}(s')$.*

Proof. Under Assumptions 20 and 21, we have that $\|\frac{\mathrm{d}}{\mathrm{d}t}\tilde{x}(t)\| = \|F(\tilde{x}(t), \tilde{u}(t))\| \leq L_F(\|\tilde{x}(t)\| + \|\tilde{u}(t)\|)$. Hence, $|\frac{\mathrm{d}}{\mathrm{d}t}\|\tilde{x}(t)\|^2| = |\langle \tilde{x}(t), \frac{\mathrm{d}}{\mathrm{d}t}\tilde{x}(t)\rangle| \leq L_F\|\tilde{x}(t)\|^2 + L_F\|\tilde{u}(t)\|\|\tilde{x}(t)\|$, which at most $\frac{3L_F}{2}\|\tilde{x}(t)\|^2 + \frac{L_F}{2}\|\tilde{u}(t)\|^2$ by the AM-GM inequality. Assumption 22 then implies this is at most $\frac{c_1}{2\alpha_Q}(Q(\tilde{x}(t)) + R(\tilde{u}(t))$, where we define $c_1 := L_F(3 + \frac{\alpha_Q}{\alpha_R})$. Thus, given any times $s_1 \leq s_2 \in [0, T]$, we have

$$\begin{aligned}\left|\|\tilde{x}(s_1)\|^2 - \|\tilde{x}(s_2)\|^2\right| &\leq \int_{t=s_1}^{s_2} \left|\tfrac{\mathrm{d}}{\mathrm{d}t}\|\tilde{x}(t)\|^2\right| \mathrm{d}t \\ &\leq \tfrac{c_1}{2\alpha_Q}\int_{t=s_1}^{s_2} (Q(\tilde{x}(t)) + R(\tilde{u}(t))\mathrm{d}t \leq \tfrac{c_1}{2\alpha_Q}\tilde{\mathcal{V}}(s_1). \end{aligned} \tag{5.12}$$

To conclude, fix a time $s \in [0, T]$. We consider two cases: **Case 1:** There is a time $\tau \in [s, T]$ such that $\|\tilde{x}(\tau)\|^2 \leq \frac{1}{2}\|\tilde{x}(s)\|^2$. Invoking (5.12),

$$\begin{aligned}\tfrac{1}{2}\|\tilde{x}(s)\|^2 &\geq \left|\|\tilde{x}(s)\|^2 - \|\tilde{x}(\tau)\|^2\right| \\ &\geq \left|\|\tilde{x}(s)\|^2 - \|\tilde{x}(s\tau)\|^2\right| \geq \tfrac{c_1}{2\alpha_Q}\tilde{\mathcal{V}}(s).\end{aligned}$$

Case 2: There is no such time τ, so $\|\tilde{x}(t)\|^2 \geq \frac{1}{2}\|\tilde{x}(s)\|^2$ for all $t \in [s, T]$. In this case, $\tilde{\mathcal{V}}(s) \geq \alpha_Q \int_{t=s}^{T} \|\tilde{x}(t)\|^2 \mathrm{d}t + \alpha_V\|\tilde{x}(T)^2\| \geq (\alpha_Q(T - s) + \alpha_V) \cdot \|\tilde{x}(s)\|^2/2$. Inverting, and combining both cases,

$$\|\tilde{x}(s)\|^2 \leq \frac{1}{\alpha_Q}\max\{c_1, \tfrac{2}{T-s+\alpha_V/\alpha_Q}\}\tilde{\mathcal{V}}(s) \tag{5.13}$$

Finally, for any $s' \in [0, s]$, arguing as in (5.12), and applying (5.13) and some simplifications (incuding $\tilde{\mathcal{V}}(s') \geq \tilde{\mathcal{V}}(s)$),

$$\begin{aligned}\|\tilde{x}(s)\|^2 &\leq \|\tilde{x}(s')\|^2 + |\int_{t=s'}^{s} (\tfrac{\mathrm{d}}{\mathrm{d}t}\|\tilde{x}(t)\|^2)\mathrm{d}t| \\ &\leq \|\tilde{x}(s')\|^2 + \frac{c_1}{2\alpha_Q}\tilde{\mathcal{V}}(s') \\ &\leq \frac{1}{\alpha_Q}\underbrace{(2c_1 + 2\tfrac{1}{T-(s')+\frac{\alpha_V}{\alpha_Q}})}_{:=\mathcal{C}_0(s')}\tilde{\mathcal{V}}(s'),\end{aligned}$$

which can be specialized to the desired cases. □

Lemma 15. *If $\tilde{u}(t)$ is an ϵ-FOS of $J_T(\cdot, x_0)$, then for each $s \in [0,T]$ we have $\tilde{\mathcal{V}}(s) \leq \alpha_Q(\mathcal{C}_1\|\tilde{x}(s)\|^2 + \frac{\mathcal{C}_2}{2}\epsilon^2)$.*

Proof. In view of Lemma 13, to obtain a bound on $\tilde{\mathcal{V}}(s)$ it suffices to bound $\mathcal{V}^{\mathtt{jac},\star}(s) := \inf_{\bar{u}_{[s,T]}} J^{\mathtt{jac}}_{T-s}(\cdot,\ \tilde{x}(s), \tilde{u}_{[s,T]})$. Moreover, we can bound $\mathcal{V}^{\mathtt{jac},\star}(s)$ by bounding $J^{\mathtt{jac}}_{T-s}(\bar{u}_{[s,T]};\ \tilde{x}(t), \tilde{u}_{[s,T]})$ for any (possibly suboptimal) control $\bar{u}_{[s,T]}$; for simplicity, let us drop the $[s,T]$-subscript going forward. We select $\bar{u}(t) = \bar{u}_1(t) + \bar{u}_2(t)$, where $\bar{u}_1(t)$ satisfies $\tilde{B}(t)\bar{u}_1(t) = -\tilde{d}(t)$, and where $\bar{u}_2$ witnesses γ-stabilizability at time s as in Assumption 24.

With this choice of $\bar{u}(t)$ the dynamics of $\bar{x}(t)$ in $J^{\mathtt{jac}}_{T-s}$ are $\frac{\mathrm{d}}{\mathrm{d}t}\bar{x}(t) = \tilde{A}(t)\bar{x}(t) + \tilde{B}(t)\bar{u}(t) + \tilde{d}(t) = \tilde{A}(t)\bar{x}(t) + \tilde{B}\bar{u}_2(t)$, and, writing out $J^{\mathtt{jac}}_{T-s}$ explicitly, we obtain

$$\mathcal{V}^{\mathtt{jac},\star}(s) \leq \int_{t=s}^{T} (Q(\bar{x}(t)) + R(\bar{u}(t))\mathrm{d}t + V(\bar{x}(T)). \tag{5.14}$$

By the elementary bound $\|\bar{u}(t)\|^2 \leq 2\|\bar{u}_1(t)\|^2 + 2\|\bar{u}_2(t)\|^2$, the following holds for constant $c = \max\{\beta_V, 2\beta_R, \beta_Q\}$,

$$\mathcal{V}^{\mathtt{jac},\star}(s) \leq 2\beta_R \int_{t=s}^{T} \|\bar{u}_1(t)\|^2 \mathrm{d}t \tag{5.15}$$

$$+ c\left(\int_{t=s}^{T} (\|\bar{x}(t)\|^2 + \|\bar{u}(t)\|^2)\mathrm{d}t + \|\bar{x}(T)\|^2\right). \tag{5.16}$$

To bound (5.16), we observe that $\frac{\mathrm{d}}{\mathrm{d}t}\bar{x}(t) = \tilde{A}(t)\bar{x}(t) + \tilde{B}(t)\bar{u}(t) + \tilde{d}(t) = \tilde{A}(t)\bar{x}(t) + \tilde{B}(t)\bar{u}_2(t)$, which corresponds to the $\hat{x}(t)$ dynamics in the definion of γ-stabilizability; thus, (5.16) is at most $c \cdot \gamma\|\tilde{x}(s)\|^2$.

To bound (5.15), we use (25) to bound $\|\bar{u}_1(t)\|^2 \leq 2L_x^2\|\tilde{x}(t)\|^2 + 2L_u^2\|\tilde{u}(t)\|^2 \leq 2c'(Q(\tilde{x}(t)) + R(\tilde{u}(t))/\beta_R$, where $c' = \beta_R \max\{\frac{L_x^2}{\alpha_Q}, \frac{L_u^2}{\alpha_R}\}$. Hence, in view of Lemma 13, and the principle of optimality:

$$\begin{aligned} 2\beta_R \textstyle\int_{t=s}^{T} \|\bar{u}_1(t)\|^2 \mathrm{d}t &\leq 4c'\beta_R \textstyle\int_{t=s}^{T} (Q(\tilde{x}(t)) + R(\tilde{u}(t))\mathrm{d}t \\ &\leq 4c'\beta_R \tilde{\mathcal{V}}(s) \leq 4c'\beta_R(\mathcal{V}^{\mathtt{jac},\star}(s) + \tfrac{\epsilon^2}{2\alpha_R}) \end{aligned}$$

Putting the bounds together and rearranging:

$$(1 - 4\beta_R c')\mathcal{V}^{\mathtt{jac},\star}(s) \leq c \cdot \gamma\|\tilde{x}(s)\|^2 + \tfrac{2c'\beta_R}{\alpha_R}\epsilon^2$$

Under Assumption 25, we have $4\beta_R c' \leq 1/2$, so that

$$\mathcal{V}^{\mathtt{jac},\star}(s) \leq 2c \cdot \gamma\|\tilde{x}(s)\|^2 + \tfrac{4c'\beta_R}{\alpha_R}\epsilon^2. \tag{5.17}$$

We recognize $2c\gamma \leq \mathcal{C}_1$, and $\frac{4c'\beta_R}{\alpha_R} = \mathcal{C}_2 - \frac{1}{2\alpha_R}$, and invoke Lemma 13 to obtain the desired bound. □

By the Fundamental Theorem of Calculus,

$$\begin{aligned} -\tfrac{\mathrm{d}}{\mathrm{d}s}\tilde{\mathcal{V}}(s) &= Q(\tilde{x}(s)) + R(\tilde{u}(s)) \geq \alpha_Q \|\tilde{x}(s)\|^2 \\ &\geq \tfrac{1}{\mathcal{C}_1}\tilde{\mathcal{V}}(s) - \tfrac{\alpha_Q \mathcal{C}_2 \epsilon^2}{2\mathcal{C}_1}, \end{aligned}$$

where the last line uses Lemma 15. Integrating the bound and again invoking Lemma 15,

$$\begin{aligned} \tilde{\mathcal{V}}(s) &\leq \exp(-\tfrac{s}{\mathcal{C}_1})\tilde{\mathcal{V}}(0) + \tfrac{\alpha_Q \mathcal{C}_2 \epsilon^2}{2\mathcal{C}_1} \int_{t=0}^{s} \exp(-\tfrac{t}{\mathcal{C}_1})\mathrm{d}t \\ &\leq \exp(-\tfrac{s}{\mathcal{C}_1})\tilde{\mathcal{V}}(0) + \tfrac{\alpha_Q}{2}\mathcal{C}_2\epsilon^2 \\ &\leq \alpha_Q \cdot (\mathcal{C}_1 e^{-\frac{s}{\mathcal{C}_1}} \cdot \|\tilde{x}(0)\|^2 + \mathcal{C}_2\epsilon^2). \end{aligned} \tag{5.18}$$

Finally, in the case where $\alpha_V > 0$, Lemma 14 lets us convert the above bound to one on $\|\tilde{x}(s)\|^2$, replacing α_Q with $\mathcal{C}_0$, as desired. In the case where $\alpha_V = 0$, for each $\delta \in [0, T]$ application of Lemma 14 yields the desired result for each $s \in [0, T-\delta]$. For each $s \in [T-\delta, T]$ Lemma 14 yields

$$\begin{aligned} \|\tilde{x}(s)\|_2^2 &\leq \mathcal{C}_0^\delta \cdot (\mathcal{C}_1 e^{-\frac{T-\delta}{\mathcal{C}_1}} \cdot \|\tilde{x}(0)\|^2 + \mathcal{C}_2\epsilon^2) \\ &\leq \mathcal{C}_0^\delta \cdot (e^{\frac{\delta}{\mathcal{C}_1}}\mathcal{C}_1 e^{-\frac{s}{\mathcal{C}_1}} \cdot \|\tilde{x}(0)\|^2 + \mathcal{C}_2\epsilon^2). \end{aligned}$$

Connection to Feedback Linearization

We begin by applying our sufficient conditions to feedback linearizable systems, perhaps the most widely studied and well-characterized class of systems in the nonlinear geometric control literature [90, Chapter 9]. Roughly speaking, a system is feedback linearizable if it can be transformed into a linear system using state feedback and a coordinate transformation. Formally, we say that (2.1) is feedback linearizable if it is both control-affine, namely, of the form

$$F(x, u) = f(x) + g(x)u,$$

where $f\colon \mathbb{R}^n \to \mathbb{R}^n$ and $g\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$, and if there exists a change of coordinates $\xi = \Phi(x)$, where $\Phi\colon \mathbb{R}^n \to \mathbb{R}^n$ is a diffeomorphism, such that in the new coordinates the dynamics of the system are of the form

$$\dot{\xi} = \hat{A}\xi + \hat{B}[\hat{f}(\xi) + \hat{g}(\xi)u] := \hat{F}(\xi, u), \tag{5.19}$$

where $\hat{A} \in \mathbb{R}^{n\times n}$ and $\hat{B} \in \mathbb{R}^{n\times m}$ define a controllable pair $(\hat{A}, \hat{B})$, $\hat{f}\colon \mathbb{R}^n \to \mathbb{R}^m$ and $\hat{g}\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$ is such that $\hat{g}(\xi)$ is invertible for each $\xi \in \mathbb{R}^n$. We will let $\hat{g}_i(\xi)$ denote the i-th column of $\hat{g}(\xi)$. We emphasize that this *global* transformation is distinct from the local Jacobian linearizations employed earlier. In this case the application of the feedback rule $u(\xi, v) = \hat{g}^{-1}(\xi)[-\hat{f}(\xi) + v]$, where $v \in \mathbb{R}^m$ is a new 'virtual' input, results in $\dot{\xi} = \hat{A}\xi + \hat{B}v$. In essence, feedback linearization reveals a linear structure underlying the global geometry of the system. Clearly $\hat{F}$ satisfies Assumption 23, and the following proposition provides sufficient conditions for Assumptions 24 and 25 to hold in the new coordinates:

Proposition 1. *Suppose that* (2.1) *is feedback linearizable, and let* $\hat{f}$, $\hat{g}$, $\hat{A}$ *and* $\hat{B}$ *be as defined above. Assume that i) there exists* $\hat{L} > 0$ *such that* $\|\frac{d}{d\xi}\hat{f}(\xi)\| < \hat{L}$ *for each* $\xi \in \mathbb{R}^n$ *and ii)* $\hat{g}(\cdot)$ *is constant on* $\mathbb{R}^n$*. Then there exists* $\gamma > 0$ *such that along each trajectory* $(\tilde{\xi}(\cdot), \tilde{u}(\cdot))$ *of* $\hat{F}$ *the associated Jacobian linearization* $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ *is* γ*-stabilizable. Furthermore the drift term* $\tilde{d}(t) = \hat{f}(\xi(t)) - \tilde{A}(t)\tilde{\xi}(t)$ *satisfies* $\|\tilde{B}^\dagger(t)\tilde{d}(t)\|_2 \leq 2\hat{L}\|\tilde{\xi}(t)\|_2$.

Remark 6. *Suppose that the representations of the state running and terminal costs,* $\hat{Q} := Q \circ \Phi^{-1}$ *and* $\hat{V} := V \circ \Phi^{-1}$*, are convex in the linearizing coordinates and satisfy pointwise bounds as in Assumption 22. Further assume that the assumptions made of* $\hat{F}$ *in Proposition 1 hold. Then the conclusions of Theorem 8 can be applied to the representation of* $J_T(\cdot, \xi_0)$ *in the linearizing coordinates by rescaling* $\hat{Q}$ *and* R *appropriately.*

Thus, the global linearizing coordinates provide a useful tool for verifying the sufficient conditions in Theorem 8. Moreover, as we illustrate with our counter-examples, they also provide insight into what goes wrong in cases where the state costs do not lead to stabilizing behavior.

While we illustrate this point further with our examples, let us briefly remark on the necessity of the conditions in Proposition 1 (for our analysis). First, note that along a given solution $(\tilde{\xi}(\cdot), \tilde{u}(\cdot))$ we have $\tilde{A}(t) = \hat{A} + \hat{B}[\frac{d}{d\xi}\hat{f}(\tilde{\xi}(t)) + \sum_{i=1}^m \frac{d}{d\xi}g_i(\tilde{\xi}(t))\tilde{u}_i(t)]$ and $\tilde{B}(t) = \hat{B}\hat{g}(\tilde{\xi}(t))$ and $\tilde{d}(t) = \tilde{A}(t)\tilde{\xi}(t) - \frac{d}{d\xi}\hat{f}(\tilde{\xi}(t))\tilde{\xi}$. When condition *ii)* is violated, even when $\frac{d}{d\xi}\hat{g}(\xi)$ can be bounded globally, the linear growth of $\tilde{A}(t)$ with respect to $\tilde{u}(t)$ may make the pair $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ more difficult to stabilize (in the sense of Assumption 24) for large values of the input. Moreover, in this case $\tilde{d}(t)$ will have quadratic cross-terms in $\tilde{\xi}(t)$ and $\tilde{u}(t)$, which may violate the growth conditions in Assumption 25. Similar issues arise when $\frac{d}{d\xi}\hat{f}(\xi)$ is not bounded globally. This occurs, for example, in Lagrangian mechanical systems wherein the Coriolis terms display quadratic growth in the generalized velocities of the system. The core challenge in each of these cases, from the perspective of our analysis, is that without making additional structural Assumptions beyond those in Proposition 1 it is difficult to rule out cases where the time-varying approximation to the dynamics along some trajectory of the system is arbitrarily difficult to stabilize.

Counterexamples

We present three counterexamples and one positive example to illustrate the necessity of our conditions and generality of our results. The first counterexample demonstrates the necessity of small control costs. The second demonstrates the necessity of the matching condition between the drift $\tilde{d}(t)$ and range of the linearized B-matrix $\tilde{B}(t)$. The third counterexample constructs an example where our conditions hold, and thus first-order stationary points are stabilizing, but where there exists a `FOS` which is *not a global optimum*. This reveals that our conditions are more generally than popular forms of "hidden convexity" such as quasi-convexity. Finally, our fourth example describes a system which is not input affine, but for which the regularity conditions outlined above still hold.

Relative Weighting of State and Input Costs We first consider the simple inverted pendulum in Figure 5.1. The states are $(x_1, x_2) = (\theta, \dot{\theta})$, where θ is the angle of the arm from vertical. The dynamics are governed by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ k \sin(x_2) + u \end{bmatrix},$$

where $k = g\ell$ with $g > 0$ the gravitational constant and $\ell > 0$ the length of the arm. The cost to be minimized is

$$J_T(\cdot, x_0) = \int_0^T \|\tilde{x}(t)\|_Q^2 + r\|\tilde{u}(t)\|_2^2 \mathrm{d}t + \|\tilde{x}(T)\|_{Q_V}^2,$$

where the scalar parameter $r > 0$ is used to control the relative weighting of the input and state costs and

$$Q_V = \begin{bmatrix} 1/4 & 1/(5\sqrt{2}) \\ 1/(5\sqrt{2}) & 1 \end{bmatrix} \quad Q = \begin{bmatrix} 1 & -1/4 \\ -1/4 & 1 \end{bmatrix}.$$

Note that the dynamics are linearizable, as they are already in the form (5.19). Moreover, applying Proposition 1, one can show that that Assumptions 22 through 25 are satisfied with parameters $\alpha_Q = \beta_Q = \frac{3}{4}$, $\alpha_R = \beta_R = r$, $L_x = k$ and $L_u = 0$. Considering $k = 10$, we find that if if $r < \frac{3}{4k^2} = \frac{3}{400}$ the sufficient conditions for exponential stability of Theorem 8 are satisfied. However, if r is not small enough then there may exists undesirable first order stationary points of the cost functional. Specifically, consider the initial condition $x_0 = (\frac{3\pi}{4}, 0)^T$ and the control signal $\tilde{u}(\cdot) \equiv -k \sin(\frac{3\pi}{4}) = -\frac{10}{\sqrt{2}}$, which generates the trajectory $\tilde{x}(\cdot) \equiv x_0$. The costate along this arc is $p(\cdot) \equiv (\frac{3\pi}{16}, \frac{3\pi}{20\sqrt{2}})$ and the gradients of the objective is given by $\nabla J_T(\tilde{u}; x_0)(t) = p(t) + ru(t)$. Thus, we see that if we choose $r = \frac{3\pi}{200}$ then we will have $\nabla J_T(\tilde{u}; x_0)(\cdot) \equiv 0$, which demonstrates that $\tilde{u}$ is a stationary point of the cost function. Thus, while RHC stability results which rely on global optimality of each planning problem predict stabilizing behavior for sufficiently large $T > 0$ [43], this example demonstrates that algorithms which only find first order-stationary points may be 'too myopic' to guarantee stability unless the input cost is small enough.

It is interesting to note that it is impossible to find a stationary pair $(\hat{x}(\cdot), \hat{u}(\cdot))$ of $J_T(\cdot, x_0)$ with the property that $\hat{x}(\cdot) \equiv x_0$ if we instead have $x_0 \in (-\frac{\pi}{2}, \frac{\pi}{2})$. Indeed, if we pick $\tilde{u}(\cdot) = -\sin(x_0)$ so that $\hat{x}(\cdot) \equiv x_0$, then in this case the costate will satisfy the differential equation $-\dot{p}(t) = \hat{A}(t)p(t) + 2q\hat{x}(t)$ where $\hat{A}(\cdot) \equiv \cos(x_0) > 0$. Thus, in this case the costate cannot be a constant function of time, which means that $\hat{u}(\cdot)$ cannot be a stationary point of $J_T(\cdot, x_0)$.

More broadly, consider a general 1-dimensional system $\dot{x} = F(x, u)$ which satisfies Assumptions 23 and 24. One can verify that and trajectory $(\check{x}(\cdot), \check{u}(\cdot))$ of F such that $\check{x}(\cdot) \equiv x_0$ and $\check{u}(\cdot) \equiv \check{u}_0$ (for some $x_0, \check{u} \in \mathbb{R}$) can only be a stationary pair of $J_T(\cdot, x_0)$ if $\frac{\partial}{\partial x} F(x_0, \bar{u}) < 0$ and $\check{d}(\cdot) \equiv \check{d}_0 = F(x_0, \check{u}_0) - \check{A}x_0 - \check{B}u_0$ is such that $sign(d_0) = sign(x_0)$, where $(\check{A}_0, \check{B}_0)$ are the Jacobian linearization of F at $(x_0, \check{u}_0)$. This highlights the need to further study how

the geometry of the Jacobian linearizations and drift terms promote or inhibit stabilizing behavior.

Structure of Local Drift Term Consider the flexible link manipulator depicted in Figure 5.1. The state is $(x_1, x_2, x_3, x_4)^T = (\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2)^T$, where θ_1 is the angle of the arm from vertical and θ_2 is the difference between the angle of the arm and the internal angle of the motor. The dynamics are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ -K_1 \sin(x_1) + K_2 x_3 \\ x_4 \\ -K_1 \sin(x_1) - (K_2 + K_3) x_3 - u \end{bmatrix},$$

where $K_1 = g\ell$, $K_2 = \frac{k}{M}$ and $K_3 = \frac{k}{I}$, where g is the gravitational constant, ℓ is the length of the arm, $k = 10$ is the spring coefficient, M the mass of the arm and I the internal inertia of the motor. For concreteness, we will assume that these physical parameters are such that $K_1 = 20$ and $K_2 = K_3 = 1$. We apply cost functionals of the form

$$J_T(\cdot; x_0) = \int_0^T \|\tilde{x}(t)\|_Q + r\|\tilde{u}(t)\|_2^2 \mathrm{d}t + \|\tilde{x}(T)\|_{Q_V},$$

$$Q := \begin{bmatrix} 1 & \frac{1}{4} & 0 & 0 \\ \frac{1}{4} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad Q_V := \begin{bmatrix} 5 & -1 & -\frac{1}{4} & 0 \\ -1 & 5 & 0 & 0 \\ -\frac{1}{4} & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}.$$

While the running and terminal costs satisfy Assumption 22 in the x coordinates, Assumption 23 is violated for this parameterization of the control system. Consider the initial condition $x_0 = (\pi, 0, 0, 0)^T$. Note that the input $\tilde{u}(\cdot) \equiv 0$ generates the trajectory $\tilde{x}(\cdot) \equiv x_0$. The reader may verify that the costate along this trajectory is $p(t) \equiv (\pi, \frac{1}{4}\pi, 0, 0)$ and the gradient at this point in the optimization space is defined by $J_T(\tilde{u}; x_0)(\cdot) \equiv 0$. Note that this is true for every choice of prediction horizon $T > 0$ and choice of the scaling parameter $r > 0$. Thus, regardless of the prediction horizon, algorithms which find first-order stationary points may get stuck at this undesirable equilibrium.

We can also see how the proposed cost function fails to guide local search algorithms to stabilizing solutions by studying its structure in a set of linearizing coordinates. Indeed, consider coordinates defined by $\xi = \Phi(x)$ where

$$\Phi(x) = (x_1, x_2, -K_1 \sin(x_1) + K_2 x_3, -K_1 \cos(x_1) x_2 + K_2 x_4).$$

These coordinates are obtained by input-output linearizing the dynamics with the output $y = x_1$ (see [90, Chapter 9]), and in the new coordinates the system can be shown to be of the form (5.19). Thus, in the new coordinates Assumption 23 is satisfied, however, Assumption 22 is not satisfied. Indeed, due to the nonlinearities in Φ, the reader may verify that the maps $z \to \|\Phi^{-1}(z)\|_Q^2$ and $z \to \|\Phi^{-1}(z)\|_{Q_V}$ are not convex. Thus, in these coordinates, the

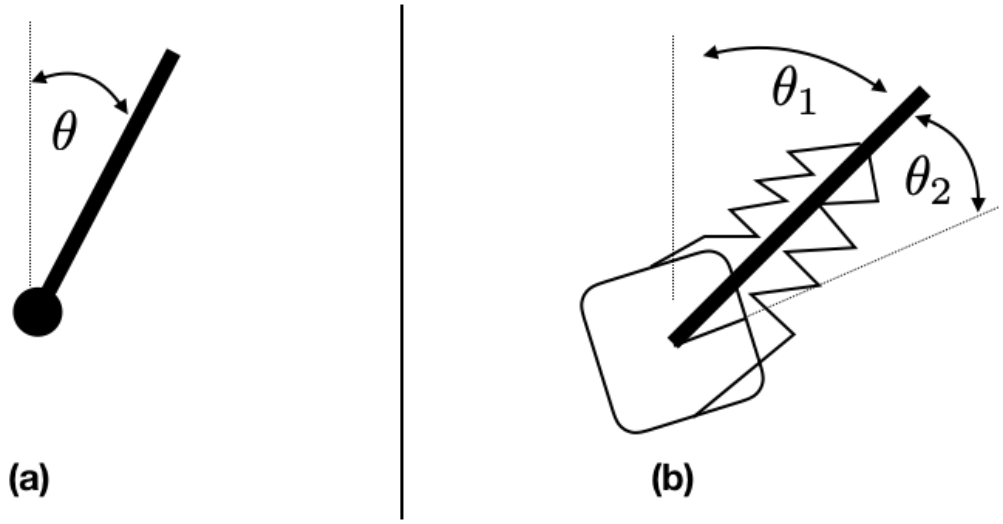

Figure 5.1: (a) Schematic for the simple inverted pendulum (b) schematic for the inverted pendulum with a flexible joint.

local structure of the state costs attracts trajectories towards the undesirable equilibrium point we identified above.

Non-Control-Affine System Consider the scalar system $\dot{x} = f(x)+bu+k\cos(x)\sin(u)$, where $f\colon \mathbb{R} \to \mathbb{R}$ satisfies $f(0) = 0$ and $\|\frac{d}{dx}\tilde{f}(x)\| < L$ for each $x \in \mathbb{R}^n$ and $b > k > 0$. Along a given system trajectory $(\tilde{x}(\cdot), \tilde{u}(\cdot))$ the Jacobian linearization is $(\tilde{A}(\cdot), \tilde{B}(\cdot))$ where $\tilde{A}(t) = \frac{d}{dx}f(\tilde{x}(t)) - k\sin(\tilde{x}(t))\sin(\tilde{u}(t))$ and $\tilde{B}(t) = b + k\cos(\tilde{x}(t))\cos(\tilde{u}(t))$. Note that $|\tilde{A}(t)| < L+k$ and $\tilde{B}(t) > b-k$, and thus it is straightforward to argue that the linearizations along the trajectories satisfy Assumption 24 for a suitable parameter $\gamma > 0$. The disturbance term associated to the chosen trajectory is $\tilde{d}(t) = f(\tilde{x}(t)) + k\cos(\tilde{x}(t))\sin(\tilde{u}(t)) - \tilde{A}(t)\tilde{x}(t) - k\cos(\tilde{x}(t))\cos(\tilde{u}(t))\tilde{u}(t)$, which can be seen to satisfy $\|B^{\dagger}(t)\tilde{d}(t)\| \leq \frac{1}{b-k}((2L+k)\|\tilde{x}(t)\| + 2k\|\tilde{u}(t)\|)$. Thus, we see that these dynamics satisfy Assumption 25. Further, if in addition we have $\frac{2k}{b-k} < \frac{1}{8}$ then Assumption 25 can be satisfied by choosing cost functions which Satisfy Assunption 22 and weight the state and input penalties appropriately. While most of the systems we consider in this chapter are control-affine, this example demonstrates that Theorem 8 can be applied to non-control-affine systems so long as the nonlinearity in the input is not 'too large'.

5.3 First-Order Stabilty Guarantees for Receding Horizon Control

In *receding horizon control* (RHC) or *model predictive control*, a planner solves $\inf_{\tilde{u}(\cdot)} J_T(\tilde{u}, x(t))$, where $x(t)$ is the current state of the real world system. The planner then applies the resulting

open loop predictive control until a new state measurement is received and the process can be repeated. As discussed above, most formal stability guarantees require that an (approximate) globally optimal solution is found for each (generally nonconvex) planning problem. Applying Theorem 8, we provide the first stability guarantees for a formal model of nonlinear RHC which only requires planning to approximate stationary points.

First-Order Receding Horizon Control

Many practical implementations of RHC use a technique known as *warm starting*, where the predictive control returned during each optimization phase is used to construct the 'initial guess' for the subsequent planning problem. This approach has proven highly effective for systems which require rapid re-planning to maintain stability [15].

To model this approach, we define the *first-order receding horizon control* strategy, denoted `FO-RHC`, as follows. First a prediction horizon $T > 0$ and a replanning interval $\delta \in (0, T]$ are chosen and a sequence of replanning times $t_k = k\delta$ for $k \in \mathbb{N}$ are defined. Next, the process takes in an initial condition of the physical system $x_0 \in \mathbb{R}^n$ and a warm-start control $\bar{u}_0 \in \mathcal{U}_T$ specified by the user. We let $(\mathbf{x}(\cdot, x_0, \bar{u}_0), \mathbf{u}(\cdot, x_0, \bar{u}_0))$ denote the resulting trajectory produced by the control scheme described below.

At each t_k for $k \in \mathbb{N}$ a warm-start routine generates an initial guess $\bar{u}_k(\cdot) = \bar{u}_k(\cdot; x_0, \bar{u}_0) \in \mathcal{U}_T$ for the problem $J_T(\cdot; \mathbf{x}(\cdot, x_0, \bar{u}_0))$; a simple choice for such a routine is presented momentarily. The local search method then optimizes the problem using the chosen initial guess, and produces the predictive control $\tilde{u}_k(\cdot) = \tilde{u}(\cdot; x_0, \bar{u}_0)$. Note that both of these quantities depend on both the initial condition of the system and the initial warm-start control specified by the user. The predictive control is constructed via $\tilde{u}_k(\cdot) = u_T^{\texttt{plan}}(\cdot, \mathbf{x}(t_k, x_0, \bar{u}_0), \bar{u}_k) \in \mathcal{U}_T$, where the map $u_T^{\texttt{plan}}$ is used to model how the chosen search algorithm selects a predictive control given for a given initial condition and warm-start input. Finally, the actual control $\mathbf{u}(t, x_0, \bar{u}_0) = \tilde{u}_k(t - t_k)$ is applied on the interval $[t_k, t_{k+1})$, and the process repeats.

Assumption 26. *We assume that, for any $\hat{x}_0 \in \mathbb{R}^n, \bar{u} \in \mathcal{U}_T$, the planned solution $\tilde{u} = u_T^{\texttt{plan}}(\cdot, \hat{x}_0, \bar{u}) \in \mathcal{U}_T$ satisfies the following two conditions with parameter $\epsilon_0 > 0$:*

1. *$J_T(\tilde{u}; \hat{x}_0) \leq J_T(\bar{u}; \hat{x}_0)$; and,*
2. *$\tilde{u}$ is an $\epsilon_0 J_T(\tilde{u}; \hat{x}_0)^{1/2}$-`FOS` of $J_T(\cdot\ ; \hat{x}_0)$.*

The rationale for the first condition is that many popular trajectory optimization methods are *descent methods*, and therefore only decrease the value of the functional J_T. The second condition is reasonable because such methods converge to approximate first-order stationary points, even for nonconvex landscapes [12]. The normalization by $J_T(\hat{x}_0, \hat{u})^{1/2}$ affords geometric stability in Theorem 9 by ensuring the optimization terminates close enough to a stationary point for each planning problem as the system trajectory approaches the origin.

It remains to specify how the warm-starts $\bar{u}_k$ are produced for $k \geq 1$. We propose selecting $\tilde{u}_k$ with δ-delay, continuing until time T, and then applying zero input:

$$\bar{u}_{k+1}(t, x_0, \bar{u}_0) = \begin{cases} \tilde{u}_k(t+\delta, x_0, \bar{u}_0) & t \in [0, T-\delta] \\ 0 & t \in (T-\delta, T]. \end{cases}$$

While more sophisticated warm-starts may be adopted in practice, the above is preferable for the present analysis because (a) it does not require further system knowledge, and (b) is ammenable to transparent stability guarantees.

Sufficient Conditions for Exponential Stability of FO-RHC

Finally, we apply Theorem 8 and its assumptions to provide sufficient conditions for the stability of FO-RHC. In order to obtain exponential convergence, we will require that, given a desired replanning interval $\delta > 0$, the prediction horizon $T > 0$ is sufficiently large and the optimality parameter $\epsilon_0 > 0$ in Assumption 26 is sufficiently small:

Theorem 9. *Let the assumptions in Theorem 8 hold. Further assume that the search algorithm chosen for* FO-RHC *satisfies the conditions in Assumption 26. Then for any prediction horizon $T > 0$, replanning interval $\delta \in (0, T]$ and optimality parameter $\epsilon_0 < \sqrt{2\alpha_Q \mathcal{C}_2}$, and each initial condition for the physical system $x_0 \in \mathbb{R}^n$ and initial warm-start decision variable $\bar{u}_0 \in \mathcal{U}_T$ the system trajectory $(\mathbf{x}(\cdot, x_0, \bar{u}_0), \mathbf{u}(\cdot, x_0, \bar{u}_0))$ generated by the corresponding* FO-RHC *scheme satisfies*

$$\|\mathbf{x}(t_k, x_0, \bar{u}_0)\|_2 \leq \sqrt{M(\delta, T, \epsilon_0)} e^{\eta(\delta, T, \epsilon_0) t_k} \|x_0\|_2,$$

for each $k \in \mathbb{N}$ where we define

$$\begin{aligned} M(\delta, \epsilon_0) &:= \mathcal{C}_0^\delta \mathcal{C}_1 \left(1 - \tfrac{\alpha_Q}{2} \mathcal{C}_2 \epsilon_0^2\right)^{-1} \\ \eta(\delta, T, \epsilon_0) &:= \tfrac{1}{2\delta} \ln \left(e^{-\delta/\mathcal{C}_1} + \mathcal{T}(\delta, T) + \mathcal{E}(\delta, \epsilon_0)\right) \\ \mathcal{E}(\delta, \epsilon_0) &:= \tfrac{1}{2} \mathcal{C}_0^\delta \mathcal{C}_2 e^{2L_F \delta} ((\delta+1)\alpha_Q + \alpha_V) \epsilon_0^2 \\ \mathcal{T}(\delta, T) &:= \mathcal{C}_0^\delta \mathcal{C}_1 (\delta \alpha_Q + \alpha_V) e^{-\frac{T}{\mathcal{C}_1} + \delta(\frac{1}{\mathcal{C}_1} + 2L_F)}. \end{aligned}$$

Proof. We first assume that $\alpha_V > 0$ and bound the rate of covegence in terms of $\mathcal{C}_0$; the steps of the proof can be repeated by replacing $\mathcal{C}_0$ with $\mathcal{C}_0^\delta$ throughout and then applying the tighter of the two bounds.

For each $k \in \mathbb{N}$ we will let $\tilde{J}_k = J_T(\tilde{u}_k(\cdot; x_0, \bar{u}_0), x_0)$ and $\tilde{J}_k = J_T(\tilde{u}_k(\cdot; x_0, \bar{u}_0), x_0)$. Respectively, these are the cost incurred by the k-th planning solution and the k-th warm-start initial guess. For simplicity, we drop the dependence on x_0 and $\bar{u}_0$ from here on. We also let $(\tilde{x}_k(\cdot), \tilde{u}(\cdot))$ and $(\bar{x}_k(\cdot), \bar{u}_k(\cdot))$ denote the corresponding system trajectories. Our goal is to show that the sequence of losses $\{\tilde{J}_k\}_{k=1}^\infty$ is geometrically decreasing. Indeed, using property 1) of Assumption 26 we have

$$\begin{aligned}\tilde{J}_T^{k+1} &\leq \bar{J}_T^{k+1} = \bar{J}_T^{k+1} - \tilde{J}_T^k + \tilde{J}_T^k \\ &= \int_{T-\delta}^{T} Q(\bar{x}_{k+1}(\tau)) + R(\bar{u}_{k+1}(\tau))d\tau + V(\bar{x}_{k+1}(T)) \\ &- \int_0^{\delta} Q(\tilde{x}_k(\tau)) + R(\tilde{u}_k(\tau))d\tau - V(\tilde{x}_k(T)) + \tilde{J}_T^k \\ &\leq \int_{T-\delta}^{T} \alpha_Q \|\bar{x}_{k+1}(\tau)\|_2^2 + \alpha_V \|\bar{x}_{k+1}(T))\|_2^2 \\ &+ \left(e^{-\frac{\delta}{\mathcal{C}_1}} + \frac{\alpha_Q}{2}\mathcal{C}_2\epsilon_0^2\right)\tilde{J}_k\end{aligned} \tag{5.20}$$

where the second inequality follows from Assumption 22, the fact that $\bar{u}_{k+1}(t) = 0$ for each $t \in [T-\delta, T]$, and applying the second inequality in (5.18) which shows that $J_{T-\delta}(\tilde{u}_k|_{[\delta,T]}, \tilde{x}_k(\delta)) \leq e^{-\frac{\delta}{\mathcal{C}_1}}\tilde{J}_k + \frac{\alpha_Q}{2}\mathcal{C}_2\epsilon_0^2\tilde{J}_k)$, where we have also used property 2) of Assumption 26. The second fact also implies that $|\dot{\tilde{x}}_{k+1}(t)| = |F(x_{k+1}(t), 0)| \leq L_F|\bar{x}(\bar{x}(t))|$ (by Assumption 21) for each $t \in [T-\delta, T]$. Thus, by a standard application of a Gronwall-type inequality for each $t \in [T-\delta, T]$ we have will will have

$$\|\bar{x}_{k+1}(t)\|_2^2 \leq e^{2L_F\delta}\|\bar{x}_{k+1}(T-\delta)\|_2^2 \tag{5.21}$$

$$\leq e^{2L_F\delta}\mathcal{C}_0\left(\mathcal{C}_1 e^{-\frac{T-\delta}{\mathcal{C}_1}}\tilde{J}_k + \frac{1}{2}\mathcal{C}_2\epsilon_0^2\tilde{J}_k\right) \tag{5.22}$$

where we have used the fact that $\tilde{x}_k(T-\delta) = \bar{x}_{k+1}(T-\delta)$, property 2 from Assumption 26 and Theorem 8. Combining the above observations with the final inequality in (5.20), integrating, and rearranging terms, and simplifying provides

$$\tilde{J}_{k+1} \leq \left(e^{-\frac{\delta}{\mathcal{C}_1}} + \mathcal{C}_0\left[\mathcal{C}_1 e^{2L_F\delta}(\delta\alpha_Q + \alpha_V)e^{-\frac{T-\delta}{\mathcal{C}_1}} + \frac{1}{2}e^{2L_F\delta}((\delta+1)\alpha_Q + \alpha_V)\mathcal{C}_2\epsilon_0^2\right]\right)\tilde{J}_k \tag{5.23}$$

Which simplifies to

$$\tilde{J}_{k+1} \leq \left(e^{-\frac{\delta}{\mathcal{C}_1}} + [\mathcal{T}(\delta, T) + \mathcal{E}(\delta, \epsilon_0)]\right)\tilde{J}_k. \tag{5.24}$$

This geometric decay implies that

$$J_T(\tilde{u}_k(0), \tilde{x}_k(0)) \leq e^{2\eta(\delta,T,\epsilon_0)t_k}J_T(\tilde{u}_0(0), \tilde{x}_0(0)) \tag{5.25}$$

This can then be converted into the desired bound on the state trajectory by applying Lemmas 14 and 15. □

To interpret the above constants, first note that for a fixed replanning interval $\delta > 0$ we have $\lim_{\epsilon_0 \to 0} M(\delta, \epsilon_0) = \bar{\mathcal{C}}_0^{\delta}\mathcal{C}_1$ and $\lim_{\substack{T\to\infty \\ \epsilon_0\to 0}} \eta(\delta, T, \epsilon_0) = \frac{1}{2\mathcal{C}_1}$. Thus, in the limiting case

FO-RHC recovers the exponential rate of convergence predicted by Theorem 8. Next, note that $\eta(\delta, T, \epsilon_0)$ will only be negative if $e^{-\frac{\delta}{C_1}} + \mathcal{T}(\delta, T, \epsilon_0) + \mathcal{E}(\delta, \epsilon_0) < 1$. Thus, for our estimate on the rate of convergence to be exponentially decaying we require that T is (at least) as big as $T \geq \ln(\bar{\mathcal{C}}_0^{\delta}\mathcal{C}_1(\delta\alpha_Q + \alpha_V)) + \delta(1 + \mathcal{C}_1 2L_F)$.

It is worth noting the significant differences between this stability result and typical stability results in the literature (e.g. [44, 33, 34]). Typical stability results use the value function $V_T(x_0) := \inf_{u(\cdot)} J_T(\tilde{u}(\cdot); x_0)$ to certify the stability of the overall receding horizon process (under the assumption that a global minimizer can be found at each planning instance), and apply the principle of optimality to argue that V_T decreases between the sampling instances. In our case the principle of optimality is cannot be applied because we are not guaranteed to obtain even locally optimal solutions. Thus, our proof technique requires demonstrating in a more direct fashion, which leverages the descent condition for the planner in Assumption 26, that the cost incurred by the planner at each sampling instance is monotonically decreasing along trajectories of the system.

5.4 Outlook and Future Work

The conditions required to ensure stability in this section are admittedly quite strong. Nonetheless, the analysis presented here offers an initial foothold for building up a geometric theory which characterizes the landscape of nonlinear optimal control problems. Even though this Chapter does not touch on learning directly, understanding the underlying structure of the cost functions used in such settings is a clear prerequisite for obtaining complexity or statistical convergence guarantees in such settings. Here, we outline a few concrete directions for future work which are currently being pursued by the author and his collaborators.

Understanding the Loss Landscape

The results presented in this Chapter make a connection to full-state feedback linearization, which in many ways is the starting point for systematically studying the structure of nonlinear systems. However, in order to expand the applicability of the results, broader classes of systems, such as minimum-phase, non-minimum-phase and differentially flat systems will need to be considered. Moreover, the applicability of the results to real-world will be greatly expanded by extending the results to cover tracking scenarios as well as situations where there are constraints on the states and inputs. Finally, the current results have a weak dependence on the structure of the terminal cost, whereas in many standard analysis [44], a terminal cost which is a also a (local) control Lyapunov function for the system can be used to shorten the prediction horizon which is required to stabilize the system.

Realtime Computational Constraints

Many applications have limited computational resources and strict run-time requirements for real-time controllers. The lack of convergence guarantees for nonlinear RHC methods has precluded their use in many safety critical scenarios. An important avenue for future work will be to leverage the current analysis to bound the number of iterations particular descent methods, such as gradient descent and I-LQR, require at each planning instance to produce a controller which will stabilize the system. Such a theory would open the possibility to apply RHC methods in application domains where end-to-end verification of controllers is the norm, as is the case in many aerospace applications.

Iterative Learning Control, Learning-based Receding Horizon Control, and Policy Optimization

In ongoing work the author is leveraging the current framework to provide practical stability guarantees from learning based receding horizon control and statistical correctness guarantees for optimization-based iterative learning control methods. Both of these approaches are based on iterative linearization of the dynamics along a nominal trajectory, and the results of this chapter can naturally be applied to such scenarios. The author is also currently using insights from this chapter as the basis for studying the convergence of policy optimization methods for nonlinear optimal control problems where a controller which stabilizes the system for every point drawn from the support of a probability distribution is desired. Thus far, convergence results are only available for the special case of linear quadratic policy optimization problems [26], wherein the simple form of the dynamics makes it possible to understand the global landscape of the policy optimization problem in a way that can be connected to the local structure of the problem along a single rollout. The hope here is that the present results can provide a basis for connecting the global and local structure of nonlinear policy optimization problems.

Chapter 6

Outlook and Future Work

Nonlinear science has always been a patchwork of perspectives which each, in their own way, help us to cage in how to conceptualize and approach the grand challenges associated with systematically controlling complex real-world systems. This is reflected in the presentation of the four previous vignettes, which each touch on different aspects of geometric control, optimal control and model-based design principles. The primary perspective developed throughout the dissertation, which is worth stating plainly again here, is that the most fruitful way forward for understanding how to systematically control complex, uncertain, nonlinear systems is to use optimal control as a unifying analysis framework and design framework. In particular, optimal control provides a 'common language' which enables us to bring in insights from geometric control, modern optimization theory and machine learning on 'equal footing'.

It is the opinion of the author that the program advocated for in this dissertation will take at least a decade to reach completion, and likely much longer. However, it is also the opinion of the author that there is fundamentally no other way forward if we wish to systematically give scalable, interpretable, and practical guarantees for nonlinear data-driven control for complex high dimensional systems. Directions for pushing forward the ideas presented in the dissertation have been provided at the end of each of the chapters. Here, a brief perspective on how these ideas will interact with real-world engineering pipeline is given.

Approximate Models for Partially Observable Environments

The work presented in this dissertation has focused largely on the 'internal' dynamics of robotic systems. Here, the word internal does not refer to zero dynamics, as in Chapter 4, but rather to the dynamics that a robot, such as a quadruped, displays when it is making contact with its environment in a known and planned way. Indeed, the most significant tacit assumption that has been made throughout the entire dissertation is that we have access to full-state measurements and that there is no estimation error. In the wild, we will need our robots to be able to respond to a vast array of dynamic interactions with different environments who's features cannot be observed directly. Retracing an example

from Chapter 1, for walking robots walking on uneven terrain it is difficult to measure even the angle of the slope that the robot is walking over, let alone build a reasonable approximate dynamics model for arbitrary ground interactions. If the ideas in this dissertation are to win-out in the long run, we will need to rethink how we are modelling robotic systems to include the local environmental interaction they experience along a trajectory that is used to improve the performance of a data-driven controller.

Learning From Human Demonstrations

Deploying autonomous robots into everyday scenarios will require them to perform intricate, difficult-to-specify tasks while working around and with humans. Recently, there has been significant interest in using human demonstrations to teach robots the skills they require to operate in these scenarios. However, reliably translating human demonstrations into a controller synthesis problem that can be easily solved is extremely challenging. This dissertation work provides an ideal foundation for tackling this problem. Much like cost functions were to specific systems, the vision here is to tailor inverse learning problems to specific systems, enabling algorithms to rapidly reconcile high-level human preferences with the physical limitations of the robot. It will be essential to pursue statistical guarantees to ensure that these approaches learn to correctly identify and respond to human intent. Solving these problems will require reconciling techniques from artificial intelligence, the cognitive sciences and data-driven control, and is essential for safely and reliably deploying autonomous robotic systems.

Data Driven Hardware and Controller Co-Design

The mechanisms our robots use to interact with the world are relatively simple when compared to the tools found in nature. This is partly due to the fact that building durable yet intricate mechanisms is extremely challenging. From the control side, the bottleneck has been coordinating numerous degrees of freedom while responding to actuator dynamics that change over time due to natural wear and tear. Consequently, in current practice we often seek a middle ground between building systems that are tractable to control and systems that can perform the intricate actions required to operate in the real world. The author intends to leverage this dissertation on scalable synthesis tools, which are designed to handle such complexities, to break this stalemate and co-design new hardware mechanisms and data-driven control strategies to provide our robots with new opportunities for sensing and interacting with the world.

What if Scaling is all You Need?

It should be considered, with a healthy amount of skepticism, that is within the realm of possibility that the 'blackbox' approach to optimal control will win out in the end as the availability of computational resources grows ever-larger and we can prepare our robots,

through extensive simulation, for the complexities of the real world. Thus, as was done throughout the production of the work presented in this dissertation, it will be essential to constantly rethink how concepts from nonlinear control can be leveraged in these scenarios, as the way in which engineering teams interact with computational approaches undergoes a constant series of changes.

Bibliography

[1] Pieter Abbeel, Morgan Quigley, and Andrew Y Ng. "Using inaccurate models in reinforcement learning". In: *International Conference on Machine Learning*. 2006, pp. 1–8.

[2] Joshua Achiam et al. "Constrained Policy Optimization". In: *International Conference on Machine Learning*. 2017, pp. 22–31.

[3] A Pedro Aguiar, Joao P Hespanha, and Petar V Kokotović. "Performance limitations in reference tracking and path following for nonlinear systems". In: *Automatica* 44.3 (2008), pp. 598–610.

[4] Notker Amann, David H Owens, and Eric Rogers. "Iterative learning control using optimal feedback and feedforward actions". In: *International Journal of Control* 65.2 (1996), pp. 277–293.

[5] Notker Amann, David H Owens, and Eric Rogers. "Predictive optimal iterative learning control". In: *International Journal of Control* 69.2 (1998), pp. 203–226.

[6] Aaron D Ames and Matthew Powell. "Towards the Unification of Locomotion and Manipulation through Control Lyapunov Functions and Quadratic Programs". In: *Lecture Notes in Control and Information Sciences* 449 (2013), pp. 219–240.

[7] Aaron D Ames et al. "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics". In: *IEEE Transactions on Automatic Control* 59.4 (2014), pp. 876–891.

[8] Zvi Artstein. "Stabilization with relaxed controls". In: *Nonlinear Analysis: Theory, Methods & Applications* 7.11 (1983), pp. 1163–1173.

[9] Martino Bardi, Italo Capuzzo Dolcetta, et al. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Vol. 12. Springer, 1997.

[10] Suneel Belkhale et al. "Model-based meta-reinforcement learning for flight with suspended payloads". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1471–1478.

[11] Richard Bellman. "Dynamic programming". In: *Science* 153.3731 (1966), pp. 34–37.

[12] Dimitri P Bertsekas. "Nonlinear programming". In: *Journal of the Operational Research Society* 48.3 (1997), pp. 334–334.

[13] Dimitri P Bertsekas and John N Tsitsiklis. "Gradient convergence in gradient methods with errors". In: *SIAM Journal on Optimization* 10.3 (2000), pp. 627–642.

[14] Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1996.

[15] Hans Georg Bock et al. "Efficient direct multiple shooting in nonlinear model predictive control". In: *Scientific Computing in Chemical Engineering II* 2 (1999), pp. 218–227.

[16] Julio H Braslavsky, Richard H Middleton, and James S Freudenberg. "Cheap control performance of a class of nonright-invertible nonlinear systems". In: *IEEE transactions on automatic control* 47.8 (2002), pp. 1314–1319.

[17] Julio H Braslavsky, Mariéa M Seron, and Petar V Kokotović. "Near-optimal cheap control of nonlinear systems". In: *IFAC Proceedings Volumes* 31.17 (1998), pp. 107–112.

[18] Arthur E Bryson and Yu-Chi Ho. *Applied optimal control: optimization, estimation, and control*. Routledge, 2018.

[19] Ching-An Cheng, Andrey Kolobov, and Adith Swaminathan. "Heuristic-guided reinforcement learning". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 13550–13563.

[20] Christine Chevallereau et al. "Rabbit: A testbed for advanced control theory". In: *IEEE Control Systems Magazine* 23.5 (2003), pp. 57–79.

[21] Chef-Seng Chow and John N Tsitsiklis. "The complexity of dynamic programming". In: *Journal of complexity* 5.4 (1989), pp. 466–488.

[22] Kurtland Chua et al. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models". In: *Advances in neural information processing systems* 31 (2018).

[23] Erwin Coumans and Yunfei Bai. *PyBullet, a Python module for physics simulation for games, robotics and machine learning*. `http://pybullet.org`. 2016–2019.

[24] Xingye Da et al. "Learning a Contact-Adaptive Controller for Robust, Efficient Legged Locomotion". In: *Conference on Robot Learning*. PMLR. 2021, pp. 883–894.

[25] Marc Deisenroth and Carl E Rasmussen. "PILCO: A model-based and data-efficient approach to policy search". In: *International Conference on Machine Learning*. 2011, pp. 465–472.

[26] Maryam Fazel et al. "Global convergence of policy gradient methods for the linear quadratic regulator". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1467–1476.

[27] Bruce Francis. "The optimal linear-quadratic time-invariant regulator with cheap control". In: *IEEE Transactions on Automatic Control* 24.4 (1979), pp. 616–621.

[28] Vincent François-Lavet, Raphael Fonteneau, and Damien Ernst. "How to discount deep reinforcement learning: Towards new dynamic strategies". In: *arXiv preprint arXiv:1512.02011* (2015).

[29] Randy Freeman and Petar V Kokotovic. *Robust nonlinear control design: state-space and Lyapunov techniques*. Springer Science and Business Media, 2008.

[30] Vladimir Gaitsgory, Lars Grüne, and Neil Thatcher. "Stabilization with discounted optimal control". In: *Systems & Control Letters* 82 (2015), pp. 91–98.

[31] Vladimir Gaitsgory et al. "Stabilization with discounted optimal control: the discrete time case". In: (2016).

[32] Robert M Gower. "Convergence theorems for gradient descent". In: *Lecture notes for Statistical Optimization* (2018).

[33] Gene Grimm et al. "Examples when nonlinear model predictive control is nonrobust". In: *Automatica* 40.10 (2004), pp. 1729–1738.

[34] Gene Grimm et al. "Model predictive control: for want of a local control Lyapunov function, all is not lost". In: *IEEE Transactions on Automatic Control* 50.5 (2005), pp. 546–558.

[35] Tuomas Haarnoja et al. "Learning to walk via deep reinforcement learning". In: *arXiv preprint arXiv:1812.11103* (2018).

[36] Tuomas Haarnoja et al. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *CoRR* abs/1801.01290 (2018). eprint: 1801.01290.

[37] Tuomas Haarnoja et al. "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". In: *International conference on machine learning*. PMLR. 2018, pp. 1861–1870.

[38] Jemin Hwangbo et al. "Control of a quadrotor with reinforcement learning". In: *IEEE Robotics and Automation Letters* 2.4 (2017), pp. 2096–2103.

[39] Jemin Hwangbo et al. "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26 (2019), eaau5872.

[40] Julian Ibarz et al. "How to train your robot with deep reinforcement learning: lessons we have learned". In: *The International Journal of Robotics Research* 40.4-5 (2021), pp. 698–721.

[41] Alberto Isidori. *Nonlinear control systems*. Springer Science & Business Media, 2013.

[42] Ali Jadbabaie and John Hauser. "On the stability of receding horizon control with a general terminal cost". In: *IEEE Transactions on Automatic Control* 50.5 (2005), pp. 674–678.

[43] Ali Jadbabaie and John Hauser. "On the stability of unconstrained receding horizon control with a general terminal cost". In: *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*. Vol. 5. IEEE. 2001, pp. 4826–4831.

[44] Ali Jadbabaie, Jie Yu, and John Hauser. "Receding horizon control of the Caltech ducted fan: A control Lyapunov function approach". In: *Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No. 99CH36328)*. Vol. 1. IEEE. 1999, pp. 51–56.

[45] Ali Jadbabaie, Jie Yu, and John Hauser. "Unconstrained receding-horizon control of nonlinear systems". In: *IEEE Transactions on Automatic Control* 46.5 (2001), pp. 776–783.

[46] Tae-Jeong Jang, Chong-Ho Choi, and Hyun-Sik Ahn. "Iterative learning control in feedback systems". In: *Automatica* 31.2 (1995), pp. 243–248.

[47] Michael Janner et al. "When to trust your model: Model-based policy optimization". In: *Advances in Neural Information Processing Systems* 32 (2019).

[48] Nan Jiang et al. "The dependence of effective planning horizon on model accuracy". In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. Citeseer. 2015, pp. 1181–1189.

[49] Ryan Julian et al. "Efficient adaptation for end-to-end vision-based robotic manipulation". In: (2020).

[50] Ryan Julian et al. "Never stop learning: The effectiveness of fine-tuning in robotic reinforcement learning". In: *arXiv preprint arXiv:2004.10190* (2020).

[51] Rudolph Emil Kalman. "A new approach to linear filtering and prediction problems". In: (1960).

[52] Christopher M Kellett and Andrew R Teel. "Results on discrete-time control-Lyapunov functions". In: *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*. Vol. 6. IEEE. 2003, pp. 5961–5966.

[53] Johannes Köhler, Melanie Zeilinger, and Lars Grüne. "Stability and performance analysis of NMPC: Detectable stage costs and general terminal costs". In: *arXiv preprint arXiv:2110.11021* (2021).

[54] J Zico Kolter and Andrew Y Ng. "Policy search via the signed derivative." In: *Robotics: science and systems*. Vol. 5. 2009.

[55] Ashish Kumar et al. "Rma: Rapid motor adaptation for legged robots". In: *arXiv preprint arXiv:2107.04034* (2021).

[56] Thanard Kurutach et al. "Model-Ensemble Trust-Region Policy Optimization". In: *International Conference on Learning Representations*. 2018.

[57] Nathan Lambert, Kristofer Pister, and Roberto Calandra. "Investigating Compounding Prediction Errors in Learned Dynamics Models". In: *arXiv preprint arXiv:2203.09637* (2022).

[58] Nathan O Lambert et al. "Low-level control of a quadrotor with deep model-based reinforcement learning". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4224–4230.

[59] Joonho Lee et al. "Learning quadrupedal locomotion over challenging terrain". In: *Science robotics* 5.47 (2020).

[60] Sergey Levine et al. "End-to-End Training of Deep Visuomotor Policies". In: *Journal of Machine Learning Research* 17.1 (Jan. 2016), pp. 1334–1373. ISSN: 1532-4435.

[61] Sergey Levine et al. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems". In: *arXiv preprint arXiv:2005.01643* (2020).

[62] Zhongyu Li et al. "Reinforcement Learning for Robust Parameterized Locomotion Control of Bipedal Robots". In: *arXiv preprint arXiv:2103.14295* (2021).

[63] B. Lincoln and A. Rantzer. "Relaxing dynamic programming". In: *IEEE Transactions on Automatic Control* 51.8 (2006), pp. 1249–1260. DOI: 10.1109/TAC.2006.878720.

[64] Bo Lincoln and Anders Rantzer. "Relaxing dynamic programming". In: *IEEE Transactions on Automatic Control* 51.8 (2006), pp. 1249–1260.

[65] Zhao Mandi, Pieter Abbeel, and Stephen James. "On the Effectiveness of Fine-tuning Versus Meta-reinforcement Learning". In: *arXiv preprint arXiv:2206.03271* (2022).

[66] David Q Mayne. "Model predictive control: Recent developments and future promise". In: *Automatica* 50.12 (2014), pp. 2967–2986.

[67] David Q Mayne and Hannah Michalska. "Receding horizon control of nonlinear systems". In: *Proceedings of the 27th IEEE Conference on Decision and Control*. IEEE. 1988, pp. 464–465.

[68] David Q Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814.

[69] Hanna Michalska and David Q Mayne. "Robust receding horizon control of constrained nonlinear systems". In: *IEEE transactions on automatic control* 38.11 (1993), pp. 1623–1633.

[70] Manfred Morari and Jay H Lee. "Model predictive control: past, present and future". In: *Computers & Chemical Engineering* 23.4-5 (1999), pp. 667–682.

[71] Rémi Munos and Csaba Szepesvári. "Finite-Time Bounds for Fitted Value Iteration." In: *Journal of Machine Learning Research* 9.5 (2008).

[72] Anusha Nagabandi et al. "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning". In: *arXiv preprint arXiv:1803.11347* (2018).

[73] Anusha Nagabandi et al. "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 7559–7566.

[74] Andrew Y Ng, Daishi Harada, and Stuart Russell. "Policy invariance under reward transformations: Theory and application to reward shaping". In: *Icml*. Vol. 99. 1999, pp. 278–287.

[75] Paavo Parmas et al. "PIPPS: Flexible model-based policy search robust to the curse of chaos". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 4065–4074.

[76] Xue Bin Peng et al. "Learning agile robotic locomotion skills by imitating animals". In: *arXiv preprint arXiv:2004.00784* (2020).

[77] Xue Bin Peng et al. "Sim-to-real transfer of robotic control with dynamics randomization". In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 3803–3810.

[78] Marek Petrik and Bruno Scherrer. "Biasing approximate dynamic programming with a lower discount factor". In: *Advances in neural information processing systems* 21 (2008), pp. 1265–1272.

[79] Elijah Polak. *Optimization: algorithms and consistent approximations*. Vol. 124. Springer, 2012.

[80] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

[81] Romain Postoyan et al. "Stability analysis of discrete-time infinite-horizon optimal control with discounted cost". In: *IEEE Transactions on Automatic Control* 62.6 (2016), pp. 2736–2749.

[82] Romain Postoyan et al. "Stability guarantees for nonlinear discrete-time systems controlled by approximate value iteration". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE. 2019, pp. 487–492.

[83] James A Primbs, Vesna Nevistić, and John C Doyle. "Nonlinear optimal control: A control Lyapunov function and receding horizon perspective". In: *Asian Journal of Control* 1.1 (1999), pp. 14–24.

[84] Quanser. *Linear servo base unit with inverted pendulum*. Apr. 2021. URL: `https://www.quanser.com/products/linear-servo-base-unit-inverted-pendulum/`.

[85] Unitree Robotics. *A1*. URL: `https://www.unitree.com/products/a1/`.

[86] Ugo Rosolia and Francesco Borrelli. "Learning model predictive control for iterative tasks. a data-driven control framework". In: *IEEE Transactions on Automatic Control* 63.7 (2018), pp. 1883–1896.

[87] Ali Saberi and Peddapullaiah Sannuti. "Cheap and singular controls for linear quadratic regulators". In: *IEEE Transactions on Automatic Control* 32.3 (1987), pp. 208–219.

[88] Peddapullaiah Sannuti. "Direct singular perturbation analysis of high-gain and cheap control problems". In: *Automatica* 19.1 (1983), pp. 41–51.

[89] PEDDAPULLAIAH Sannuti and H Wason. "Multiple time-scale decomposition in cheap control problems–singular control". In: *IEEE transactions on automatic control* 30.7 (1985), pp. 633–644.

[90] Shankar Sastry. *Nonlinear systems: analysis, stability, and control*. Vol. 10. Springer Science & Business Media, 1999.

[91] Sosale Shankara Sastry and Alberto Isidori. "Adaptive control of linearizable systems". In: *IEEE Transactions on Auto. Control* 34.11 (1989), pp. 1123–1131.

[92] Angela P Schoellig, Fabian L Mueller, and Raffaello D'andrea. "Optimization-based iterative learning for precise quadrocopter trajectory tracking". In: *Autonomous Robots* 33.1 (2012), pp. 103–127.

[93] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[94] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[95] Pierre OM Scokaert, David Q Mayne, and James B Rawlings. "Suboptimal model predictive control (feasibility implies stability)". In: *IEEE Transactions on Automatic Control* 44.3 (1999), pp. 648–654.

[96] Rodolphe Sepulchre, Mrdjan Jankovic, and Petar V Kokotovic. *Constructive nonlinear control*. Springer Science & Business Media, 2012.

[97] Mariéa M Seron et al. "Feedback limitations in nonlinear systems: From Bode integrals to cheap control". In: *IEEE Transactions on Automatic Control* 44.4 (1999), pp. 829–833.

[98] Laura Smith et al. "Legged Robots that Keep on Learning: Fine-Tuning Locomotion Policies in the Real World". In: *arXiv preprint arXiv:2110.05457* (2021).

[99] Eduardo D Sontag. "A 'universal' construction of Artstein's theorem on nonlinear stabilization". In: *Systems and Control Letters* 13.2 (1989), pp. 117–123.

[100] Chen Tessler and Shie Mannor. "Reward Tweaking: Maximizing the Total Reward While Planning for Short Horizons". In: *arXiv preprint arXiv:2002.03327* (2020).

[101] Tyler Westenbroek et al. "Combining model-based design and model-free policy optimization to learn safe, stabilizing controllers". In: *IFAC-PapersOnLine* 54.5 (2021), pp. 19–24.

[102] Tyler Westenbroek et al. "Reinforcement Learning with Simple Models and Low-Level Feedback Controllers". In: *arXiv* (2022).

[103] Xiaobin Xiong and Aaron D Ames. "Dynamic and versatile humanoid walking via embedding 3d actuated slip model with hybrid lip based stepping". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 6286–6293.