Numerical Nonlinear Robust Control with Applications to Humanoid Robots

CMU-RI-TR-15-17

Jiuguang Wang

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Robotics
July, 2015

Thesis Committee: Professor Christopher G. Atkeson, chair
Professor Koushil Sreenath
Professor Bruno Sinopoli
Professor Stefan Schaal (USC)

Abstract

Robots would be much more useful if they could be more robust. Systems that can tolerate variability and uncertainty are called robust and the design of robust feedback controllers is a difficult problem that has been extensively studied for the past several decades. In this thesis, we aim to provide a quantitative measure of performance and robustness in control design under an optimization framework, producing controllers robust against parametric system uncertainties, external disturbances, and unmodeled dynamics. Under the $\mathcal{H}_\infty$ framework, we formulate the nonlinear robust control problem as a noncooperative, two-player, zero-sum, differential game, with the Hamilton-Jacobi-Isaacs equation as a necessary and sufficient condition for optimality. Through a spectral approximation scheme, we develop approximate algorithms to solve this differential game on the foundation of three ideas: global solutions through value function approximation, local solutions with trajectory optimization, and the use of multiple models to address unstructured uncertainties. Our goal is to introduce practical algorithms that are able to address complex system dynamics with high dimensionality, and aim to make a novel contribution to robust control by solving problems on a complexity scale untenable with existing approaches in this domain. We apply this robust control framework to the control of humanoid robots and manipulation in tasks such as operational space control and full-body push recovery.

Dedication

In memory of Mike Stilman.

Acknowledgements

I would like to thank the many people who have helped me along the way.

I'm very grateful to my advisor, Chris Atkeson, for his vision and guidance, without which this work would not have been possible. Member of my thesis committee, Koushil Sreenath, Bruno Sinopoli, and Stefan Schaal gave me an extremely diverse perspective for control theory and machine learning, as well as provided many valuable comments and suggestions during the editing of this manuscript.

My mentors at Georgia Tech made a tremendous impact in my academic career. Magnus Egerstedt took me in as an undergrad and dazzled me with the beauty of mathematics and systems theory. Ayanna Howard gave me the freedom to tinker with robots in her lab and made me a better engineer. Mike Stilman inspired me with his passion for humanoid robots and encouraged me to venture outside of my comfort zone.

I've benefited greatly from my interactions with many faculty & staff members, labmates, officemates, and other colleagues at CMU and Georgia Tech. I am humbled by all the support that they have given me over the course of my academic career.

I thank my family for the continuous support they have given me throughout my time in graduate school.

This work was supported in part by the National Science Foundation (ECCS-0824077, IIS-0964581, IIS-1017076), the DARPA M3 program, and the DARPA Robotics Challenge (HR0011-14-C-0011). From 2010-2013, I was also supported by the NSF Graduate Research Fellowship Program, Grant #0750271.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Motivation

Robots would be much more useful if they could be more robust. Systems that can tolerate variability and uncertainty are called robust and modern robots can rarely be classified as such. In this thesis, we aim to provide a quantitative measure of performance and robustness that leads to an optimization framework, from which a controller can be designed to robustly control the behaviors of humanoid robots. We develop these algorithms on the foundation of three ideas: global solutions through value function approximation, local solutions with trajectory optimization, and the use of multiple models to address unstructured uncertainties. Our goal is to introduce practical algorithms that are able to able to address complex system dynamics with high dimensionality.

The key design philosophy of this thesis is the idea of *design driven* robustness, as opposed to *data driven* approaches commonly found in adaptive and learning systems. In this work, we explicitly formulate sources of uncertainty, both internal and external to the system dynamics, and achieve robustness in a *top-down* manner. This is distinct from the machine learning literature, where uncertainties are identified and learned from data, processed *bottom-up*. These two schools of design are not incompatible with one another and this work aims to complement existing data driven approaches to robust control.

The main application area for this thesis is the control of complex mechanical systems,

such as humanoid robots. Humanoid robots are designed to imitate the manipulative, locomotive, perceptive, communicative, and cognitive abilities of humans [3]. On one hand, humanoid robots are the ideal platform to introduce to human domains since they are compatible with made-for-human tools and environments. On the other hand, any humanoid robot with sufficient physical capabilities poses a significant *safety risk* to nearby humans. Dynamically balancing bipedal humanoid robots are particularly dangerous due to their high center-of-mass and comparably small polygon of support. While they are more versatile on different types of terrains compared to wheeled or multi-legged robots, *robust* bipedal locomotion has not been demonstrated outside of the laboratory setting as *stability* can easily be influenced by a number of factors. Before we can expect to move these robots beyond the settings of research and into human populated environments, the issue of robustness must be addressed and it is an important component in *physical human-robot interaction*.

For a humanoid robot with many degrees-of-freedom, designing behaviors by hand is a difficult, if not intractable, task. While there have been attempts to use motion capture [4] to extract or transfer human motion to robot control, the significant differences in the kinematic structures of humans and humanoid robots make this approach difficult to apply in practice. Instead, a philosophy for the control of humanoid robots is *optimization*. By defining appropriate optimization objectives in conjunction with a model of the system dynamics and associated constraints, *emergent behaviors* can be generated automatically.

While optimization and optimal control theory have been widely applied in humanoid robot control, it is not without drawbacks. A blind application of optimization tends to yield extremal solutions that exploit the given model to improve performance. However, roboticists rarely have models that characterize the response of the robot with absolute accuracy, especially given a complex, unstable, high degree-of-freedom humanoid. Typically, the theoretical dynamics of the robot are obtained through the principles of mechanics, with experimental procedures that validate the basic equations of motion and refine the

parameters. These experiments are corrupted by noise and the identified system parameters are only known with *limited degrees of accuracy*. Hydraulic robots, in particular, have oil seal friction and leakage, which causes parameters to vary with each movement. Over time, degradation and wear also alter the parameters of the system, and unless the robot is calibrated periodically to re-identify the parameters, the model can deviate significantly from reality.

Furthermore, as humanoid robots are introduced to cluttered and uncontrolled human environments, there are various external factors that can adversely affect the system. The robot is limited by its perceptual capabilities and is vulnerable to changes in the terrain as well as actions of human workers nearby. Since *external disturbances* such as being pushed cannot be predicted accurately, it is crucial that the safety performance of the robot be guaranteed for worst-case scenarios. Contact properties, similar to seal friction, change with every move. Together, addressing the accuracy of the models and associated imperfections, as well as external disturbances, will ensure that the performance of theoretically optimal designs will be maintained in practice.

In control theory, *robust control* is an active field which addresses many issues in designing feedback controls to account for model uncertainties and external disturbances. Historically, robust control theory is rooted in analysis in the frequency domain that is specific to the control of linear dynamic systems. While recent works in this area have began to address the control of nonlinear systems in the time domain, most rely on analytically manipulating simple system dynamics to design feedback controls, which render them inapplicable to more complex systems. For a general class of nonlinear systems, necessary and sufficient conditions can be obtained in terms of the *Hamilton-Jacobi-Isaacs equation* (HJI), a partial differential equation that is difficult to solve and does not admit a smooth analytical solution in general. Numerical techniques based on *dynamic programming* solve the HJI by discretizing the state space of the system on a grid, but suffer from the *curse of dimensionality* and require extensive computational resources to pre-compute policies

offline.

In the next section, we review existing methods for nonlinear robust control and discuss how they relate to the proposed approach.

1.2 Background and related work

1.2.1 Classical robust control

In control theory, work on robust control originated in the 1970s, motivated by the invention of the Linear-Quadratic Gaussian (LQG) controller. An extension of the Linear-Quadratic Regulators (LQR), LQG models linear systems disturbed by additive Gaussian noise, subject to quadratic costs. LQG explicitly model the disturbance and should have superior performance compared to LQR, but in fact, it was shown by Doyle [5] that LQG techniques provide no global system-independent guaranteed robustness properties. The LQG case showed the need for a systematic way of demonstrating robustness margins for control designs and incorporating them into controller synthesis.

The $\mathcal{H}_\infty$ *optimal control* framework was developed in response to the need of addressing modeling errors and the *worst-case* responses of a system. Instead of optimizing the usual performance metrics such as rise time, settling time, energy, etc, $\mathcal{H}_\infty$ control focuses on sensitivity minimization, which keep the response of a system under bounded disturbances. Work by Zames [6] was representative in the formulation of sensitivity reduction by feedback in the $\mathcal{H}_\infty$ framework, and was extended by Francis [7]. The core idea in these studies is the $\mathcal{H}_\infty$-norm, a measure of the worst-case response of a system seen as the highest gain value on a Bode magnitude plot. Minimizing the $\mathcal{H}_\infty$ norm is therefore an optimization problem, and when combined with traditional feedback designs, constitute a coherent framework in which controller behaviors can be guaranteed. For the history of

$\mathcal{H}_\infty$ control and a comprehensive review, we refer the readers to several well-known textbooks in this area by Bhattacharyya [8], Zhou [9], Skogestad [10], and Dullerud [11], as well as the references therein.

While classical robust control theory in the form of $\mathcal{H}_\infty$ control has been well-established, it is not directly applicable to humanoid robots since traditional $\mathcal{H}_\infty$ designs rely on frequency domain analysis techniques for linear systems, while the dynamics for humanoids are highly nonlinear. Nonlinear systems do not have properties such as superposition, scaling, and homogeneity [12], found in their linear counterparts, which means that tools from linear algebra such as the Laplace transform cannot be used. Analysis of nonlinear systems are made more difficult by phenomenon such as finite escape time, multiple isolated equilibria, limit cycles, and chaos. Control design for nonlinear systems are synthesized entirely in the time domain, which require different techniques to derive the feedback control law, particularly when the issue of robustness is considered. While the $\mathcal{H}_\infty$ norm can be formulated in the time domain and its minimization remains an optimization problem, the solution is much more difficult to obtain due to the nonlinear nature of the systems involved.

1.2.2 Dynamic programming

We began this review of modern nonlinear robust control with the *dynamic programming principle* developed by Bellman [13]. The *Bellman equation*, a central result in dynamic programming, is a necessary and sufficient condition for optimality which discretizes and solves a given optimization problem recursively. The use of the Bellman equation in a continuous-time nonlinear optimal control problem [14] results in the *Hamilton-Jacobi-Bellman (HJB) equation* [15], which applies to the most general form of nonlinear dynamics and constraints. The HJB is a nonlinear first-order partial differential equation, and when solved over the entire state space of a system, is a necessary and sufficient condition for

optimality. The solution to the HJB, the *value function*, gives the optimal *cost-to-go* for a given system and there is a relationship between the value function of an optimal control problem and the feedback control law.

In the context of nonlinear robust control with an external disturbance, the optimization problem constitutes a minimax problem in the sense that we try to minimize the objective function under the worst possible disturbances or parameter variations, which maximizes the same objective function. The minimax problem also has a game theoretic interpretation as a *two-player, noncooperative, zero-sum differential game* [16]. This problem is a game in that a minimizing player seeks to determine control inputs that minimize the objective function, while an adversarial maximizing player seeks to maximize the same objective function. It is a *differential game*, since both players are constrained by the system dynamics, a set of ordinary differential equations, as well as associated constraints. Furthermore, the game is *non-cooperative* as each player makes decisions independent of the other player. The outcome is *zero-sum*, since one player's gain is exactly balanced by the other player's loss.

In the context of game theory, the equilibrium strategy for this differential game constitutes a *Nash equilibrium* [17], also called a *saddle-point*, which can be intuitively explained as a pair of strategies secured against any attempt by one player to unilaterally change his strategy. In a continuous-time setting, the application of the dynamic programming principle to a differential game results in the *Hamilton-Jacobi-Isaacs (HJI) equation* [15], which can be seen as the dual to the HJB. It is also a first-order nonlinear partial differential equation and serves as the necessary and sufficient condition for optimality. We refer the reader to Başar [18], Engwerda [19], and Friesz [20] for a detailed treatment of differential games in the context of optimization and optimal control. For subsequent discussions, we will use the terms $\mathcal{H}_\infty$ robust control, minimax control, and differential games interchangeably. We will mostly ignore more detailed classifications of differential games as solution approaches tend to be valid for all forms of differential games with minor modifications.

While the formulation of the nonlinear robust control problem is straightforward, the difficulty in solving the HJI equation remains the biggest bottleneck to the practical application of nonlinear $\mathcal{H}_\infty$ control. For general nonlinear systems, no smooth analytical solution can be expected. One approach to solving the HJI is the notion of viscosity solution [15]. When written in the form of linear differential operators, the HJI satisfy an *ellipticity* condition on the monotonicity in the diffusion operator. As a result, existence and uniqueness of viscosity solutions can be demonstrated in the form of a value function. However, the viscosity approach can only be applied to a small subset of problems and general nonlinear solutions remain intractable. Methods from numerical analysis, such as Galerkin's method, can be used to convert the HJI from a continuous operator form to a discrete problem. Representative references in this area include Georges [21], Beard [22] [23] [24], Alamir [25], and Ferreira [26]. The drawback of Galerkin-based approaches is the need to successively produce discrete forms, which is difficult to implement in practice. Related approximate methods such as Tsiotras [27] considered the use of Taylor series approximation of the value function to iteratively solve the HJI term by term, provided that a solution to the linearized model of the nonlinear system exists. Similarly, Grimble [28] proposed a version of the $\mathcal{H}_\infty$ design in which the system is modeled as a combination of a linear and a nonlinear subsystem, where the plant can have severe non-smooth nonlinearities but the disturbance and reference models are assumed linear. These approaches do not have a closed form solution and are not guaranteed to converge.

More recently, Feng [29] [30] proposed an iterative algorithm to solve the HJI for a broad class of nonlinear system by solving a sequence of HJBs whose solutions can be approximated recursively by existing methods, with a local quadratic rate of convergence. However, as the solution of HJBs remain difficult to obtain, this approach is only the beginning of a potential direction to solve nonlinear robust control problems. Finally, more heuristic approaches have been proposed by Kim [31], which augments a nominal control design using a PID-type auxiliary input, which adds $\mathcal{H}_\infty$-type performance measures to

achieve additional robustness.

There exists a rich literature on the solution to the nonlinear $\mathcal{H}_\infty$ problem, in control theory, differential game, Hamilton-Jacobi theory, and various other fields. We refer the readers to Helton [32]and Aliyu [33], as well as the references therein, for an overview of various techniques in this area. To date, no effective analytical solutions to the HJI exists. Instead, we focus on reviewing *numerical solutions* to the HJI, and also other *approximate solutions* to $\mathcal{H}_\infty$ control which do not rely on solving the HJI.

The chief numerical algorithms for dynamic programming problems are policy iteration (also known as Howard's algorithm) and value iteration (also known as backward induction) [34]. These algorithms play an important role in the study of *Markov decision processes (MDPs)* [35] and *reinforcement learning (RL)* [36]. When solved on a grid over some bounded region of the state-space, the number of grid points grows exponentially to the dimension of the state-space. Few successful implementations of these numerical schemes have been demonstrated for systems beyond a few states.

1.2.3 Approximate dynamic programming

An extension to the numerical approaches to dynamic programming is *approximate (or adaptive) dynamic programming (ADP)*. ADP encompasses an umbrella of methods related to dynamic programming, called forward / incremental / iterative / heuristic / neurodynamic programming, and other variations. We refer readers to Powell [37] for a comprehensive review of these methods. In short, the idea of *value function approximation* is central to ADP, where the true Bellman value function is replaced by a statistical approximation followed by modifications to the iterative procedure that updates this approximate function. A related method to value function approximation is policy search, where the parameters in the parametric representation of the policy is searched directly for a solution that satisfies the control objectives. A mixture of value function approximation

and policy search is often referred to as an actor-critic design [38] in the sense that an actor directly optimizes the controller while a critic observes the performance of a given controller based on a particular value function and then derives an improved controller by updating the value function. When an explicit representation of both the policy and the value function is maintained, then both the actor and critic act simultaneously in an actor-critic design. The actor-critic architecture is widely used in *reinforcement learning* and we refer the readers to Busoniu [39] for a recent survey on the use value function approximation in this area.

There exists an extensive literature of using ADP to control dynamic systems, particularly using neural networks as function approximators. This stems from the fact that any feedforward network with a single hidden layer containing a finite number of neurons (in the simplest case, a multilayer perceptron) can approximate any continuous functions with mild assumptions on the activation function [40]. While there are other alternative universal approximators (for example, support vector machines [41]), neural networks remain popular for optimal control problems due to the availability of formal convergence properties for certain configurations. We refer the readers to Miller [42] for a comprehensive review of neural network methods for optimal control problems and Zhang [43] for ADP-specific methods. We focus the subsequent discussion on ADP for $\mathcal{H}_\infty$ design and two-player differential games, with ADP predominantly take the form of actor-critic architecture. Approaches have been proposed for systems of various configurations, including linear and nonlinear dynamics that are known, partially known, or unknown. Some approaches account for constraints on the dynamics, while others have the ability to learn in an online fashion.

The simplest application of ADP is for linear $\mathcal{H}_\infty$ control with known dynamics. Al-Tamimi [44] proposed a heuristic dynamic programming approach where action and critic networks were used to solve for the value function and the costate of a linear discrete-time zero-sum game. The approach can be seen as a way of solving the Riccati equation

that arises from the linear game. Li [45] extended this approach to achieve model-free learning assuming the underlying dynamics to be a completely unknown linear continuous-time system. In this approach, one critic network and two action networks were used to approximate the value function, control and disturbance policies, with a least squares method for estimating the unknown parameters.

Similar actor-critic methods can be applied to nonlinear systems as well. Representative approaches include Liu [46], where three neural networks were used to approximate the value function, control input, and disturbance, respectively, and updated using a greedy heuristic dynamic programming algorithm in conjunction with discrete-time affine nonlinear dynamics. Convergence can be proven for the scheme and tracking control can be accomplished through a system transformation. Similarly, Zhang [47] proposed an iterative ADP method with four action networks and two critic networks. Other methods have focused on simplifying the architecture by reducing the number of function approximations. In approaches such as Dierks [48], a single approximator based scheme was used to achieve optimal regulation and tracking control for an affine system. Zhang [49] used single critic network for each player instead of the actor-critic dual network for a non-zero-sum game.

While the most common forms of the HJI do not account for constraints on the states and control inputs, some ADP approaches can be modified to include these constraints. Representative methods include Abu-Khalaf [50], where a neurodynamic programming algorithm in conjunction with a two-player policy iteration was applied to a nonlinear affine systems with input saturations. The result is a closed-form representation of the feedback strategies and the value function. Related work by Abu-Khalaf [51] solved the HJI by decomposing it into a sequence of differential equations linear in the cost for which closed-form solutions are easier to obtain. The approach is combined with neural network approximations and policy iteration to derive numerical solutions.

While most ADP approaches are executed offline to converge to a value function and

a corresponding policy, several approaches promised the ability to identify solutions in an online setting. Assuming completely known dynamics, Vamvoudakis [52] proposed a method to learn online, in real time, an approximate local solution to the HJI. Termed synchronous ZS game policy iteration, this approach was built on the actor-critic design and used simultaneous continuous-time adaptation of critic, actor, and disturbance neural networks. With the ability to learn online, it is then natural to extend $\mathcal{H}_\infty$ designs with dynamics that are only partially known or completely unknown. Notable approaches include Mehraeen [53] with an iterative approach using neural networks that approximated a linear value function for discrete-time nonlinear systems with partially unknown internal system dynamics, approximated by an additional neural network. Zhang [54] presented the learning of a completely unknown nonlinear system in a data-driven approach, where the dynamics is essentially replaced by available input-output data. In Wu [55], an online simultaneous policy update algorithm was used for a system with unknown dynamics. Finally, Luo [56] proposed an off-policy reinforcement leaning design to learn the solution of the Issacs equation from data, under a neural network based actor-critic structure.

Since no approximation strategy works for all optimization problems without modification, ADP-based designs are highly domain specific. While most existing applications of ADP in robust control utilized neural networks due to their universal function approximation ability, these approaches are not without drawbacks. In practice, while a single hidden layer neural network can approximate any function, it is often inefficient. This is due to the fact that neurons may need to be allocated to every small volume of the input space. As the number of such small volumes grows exponentially in the dimensionality of the input space, convergence is rendered exponentially slow.

A remedy for this problem exists in machine learning in the form of so-called *deep neural networks* or *deep learning* [57], where a many-layered feedforward neural network is trained one layer at a time, treating each layer as an unsupervised restricted Boltzmann machine, then using supervised backpropagation for fine-tuning. Though a deep architecture is not a

universal approximator unless it is also exponentially large, it remains a viable alternative to traditional neural networks for many applications. So far, these deep and recurrent architectures have been used for optimal control [58], but not applied to robust control.

Reinforcement learning and learning-based solve the same robust control problem, but using a different underlying philosphy. Robust control, as formulated under the $\mathcal{H}_i nfty$ approach, explicitly models potential sources of uncertainties, both internal and external to the system. This is a *design-drive*, or *top-down* approach. Learning-based approaches, on the other hand, elect to assume no existing knowledge of the underlying dynamics, and instead explores the state space by perturbing the system. This is a *data-driven*, or *bottom-up* approach. When the level of uncertainties in the system is small and isolated, then a design-driven approach can often simply the design. If there are extensive uncertainties that are hard to model, then a data-driven approach would have an advantage.

1.2.4 Local value function approximations

The philosophy of most ADP-based techniques reviewed in Section 1.2.3 is to use universal function approximators, which are able to converge to the true value function given enough computational resources. An alternative approach is to limit the order of the approximated value function in order to gain computational efficiency.

In this approach, a simple way to approximate the value function is by defining polynomial basis functions and form an optimization problem to search for the associated coefficients. This is still done in the context of the Bellman equation, which can be relaxed to an inequality, which the polynomial approximation must satisfy. This, by definition, provides an underestimate of the value function. An example of this strategy can be found in Wang [59], which utilized a quadratic approximation in its ADP implementation. When viewed in the context of general low-order approximations to the value function, the spectrum of techniques known as *sum-of-squares (SOS) optimization* and *differential dynamic*

programming (DDP) can be classified under the broad family of ADP algorithms, though with the distinction of producing only locally optimal solutions.

In SOS, convex optimization [60] is used to automatically search for *Lyapunov functions* to prove the stability of nonlinear dynamic systems in a polynomial form. SOS is based on the idea of positivity of polynomials, which was shown by [61] to form a convex *semidefinite programming (SDP)* problem [62]. Since a Lyapunov function is a matrix polynomial with constraints on positivity (or non-negativity), SOS is a natural tool in the verification of Lyapunov function candidates. An associated problem is the estimation of the *domain of attraction (DoA)*, which defines a neighborhood around an equilibrium for which all trajectories converge to the equilibrium. The sub-level set of a Lyapunov function is related to an inner estimate of the size of the domain of attraction.

Since a value function satisfies all the properties of a Lyapunov function, the Lyapunov function found through SOS optimization can be seen as a *quadratic approximation* to the value function. While originally developed only for the verification of stability, SOS methods were extended to controller syntheses as well. The simultaneous search for a controller and the associated domain of attraction can be formulated as a single SDP problem, which can be achieved by rewriting conditions from basic Lyapunov stability theory [12] as *linear matrix inequalities (LMIs)* [63]. By assuming a *quadratic* form of the Lyapunov function, the resulting problem is *convex* and can be readily solved using *interior point methods* [64]. The reader is referred to Chesi [65] for a comprehensive review of SOS techniques for traditional control applications and we focus the subsequent discussions on results relevant to robust control.

There exists a rich literature on the use of SOS and LMIs for the control of uncertain systems and we refer the reader to a recent survey by Petersen [66] on the subject. There is, however, a distinction between the worst-case uncertainty assumed by $\mathcal{H}_\infty$ designs and the *bounded uncertainty* models used by SOS approaches such as Ichihara [67] and Ma

[68]. Where these approaches assumed a bounded uncertainty (in terms of disturbances or parametric uncertainty), they do not model the worst-case and hence are not subject to a minimax or differential game interpretation. Most applications of LMIs to $\mathcal{H}_\infty$ control focus on linear systems, such as in Gahinet [69], who introduced solvability conditions involve Riccati inequalities. Poznyak [70] proposed a class of attractive ellipsoids methods to construct linear-type feedback control synthesis that can also be applied to nonlinear affine systems in the presence of uncertainties.

For general nonlinear systems, a convex parameterization of nonlinear $\mathcal{H}_\infty$ control was given by Lu [71] in terms of nonlinear matrix inequalities, which connected the solution to Lyapunov stability but did not provide a computationally tractable solution. Further developments by Prempain [72] formulated the $\mathcal{L}_2$-analysis problem which bypassed the need to solve the HJI with a new problem linear in the Lyapunov function parameters. This was done under the SOS framework with convex state-dependent LMIs for for an affine nonlinear system. For single-input polytopic systems, Wu [73] proposed necessary and sufficient conditions for verifying robust storage functions and provided conditions for the existence of continuous robust state feedback controllers. More recently, Wei [74] utilized SOS for $\mathcal{L}_2$-gain analysis and synthesized state feedback $\mathcal{H}_\infty$ control iteratively based on an initial controller that can stabilize the system to a finite $\mathcal{L}_2$-gain. Löfberg [75] developed robust convex programming where uncertain semidefinite and second order cone constraints were employed to optimize against the worst-case cost. Finally, Zheng [76] reformulated the HJI inequalities as SDP conditions and used high order Lyapunov functions in conjunction with SOS programming to obtain output feedback laws for a class of nonlinear systems without orthonormal and decoupling assumptions.

SOS techniques are only applicable to polynomial systems. For systems dynamics that are not originally polynomial, for example, mechanical systems with trigonometric terms, a high-order Taylor expansion can be used to obtain a polynomial representation. This creates additional computational burden and creates inaccuracies in the results. In general,

computational efficiency with respect to dimensionality is a main drawback of SOS-based approaches. While SOS is not subject to the curse of dimensionality in the traditional sense, the search for a Lyapunov function remains difficult when the dimensionality of the system is high. This stems from the fact that the number of monomial basis functions used to construct the Lyapunov function explodes exponentially with the number of states. While there are techniques to actively prune unnecessary monomials and reduce the number of optimization parameters (for example, the method of Newton polytopes [77]), it leads to a trade-off with the quality of the resulting approximation. This is a criticism that can be extended to all forms of function approximators, such as those reviewed in Section 1.2.3.

More recently, several approaches have combined forms of value function approximation with *trajectory optimization* methods, which produce locally optimal solutions to the underlying optimization problem. Methods based on *differential dynamic programming (DDP)* [78], for example, have been applied to solve optimal control problems [79], with some variants [80] specific for $\mathcal{H}_\infty$ designs. DDP approximates the value function using locally quadratic models and uses local trajectory optimizers to globally optimize a policy and associated value functions. Another related class of method is the so-called path integral controls [81] [82], which utilizes an exponential transformation of the Bellman value function to reduce the stochastic Hamilton-Jacobi-Bellman equation into a linear form, which can then be solved with Monte Carlo sampling. Path integrals and related formulations using direct numerical optimization [83] can be seen as a trajectory-based value function approximation. Finally, LQR-trees [84] [85] combines SOS-based stability verification with sampling-based planners under the rapidly-exploring random tree paradigm [86], which creates a sparse tree of trajectories that probabilistically covers the entire controllable subset of state space. DDP, path integrals, and LQR-trees all have variants which deal with optimal and stochastic control problems, but so far, there is only limited work on investigating the applicability of these approaches in robust control problems.

1.2.5 Trajectory optimization

Dynamic programming (and by extension, ADP)-based approaches to optimal and robust control yield globally optimal solutions, by the virtue of solving the full necessary and sufficient conditions for optimality. As these approaches become intractable for large scale problems, researchers have turned to more relaxed forms of optimization which produce only *local solutions*. A representative approach is *Pontryagin's minimum (also maximum) principle (PMP)* [14], which is only a necessary condition for optimality when satisfied along a trajectory. Pontryagin extended the classical *calculus of variations* [87] to consider inequality constraints in the control inputs, and in particular, to cases where the control inputs are bounded.

In short, the PMP stated that the *Hamiltonian* must be minimized over the set of all permissible controls within their bounded region at each point along the path. This is an observation consistent with dynamic programming, which minimizes the Hamiltonian at each point in the state space. The PMP also defined the relationship of the state variable with the *costate*, also known as the *adjoint*, which are the Lagrange multipliers used in the constrained optimization. For simple problems, the optimal controls can be solved as functions of the costate. For more complex problems, the solution of the costate equations and the associated optimal controls form a *two-point boundary-value problem (BVP)* [88], which can be solved numerically to yield an open-loop trajectory to the underlying optimization problem. We refer the reader to Bryson [14] and Kirk [89] for a classical treatment of the PMP in optimal control.

The PMP has been extended to differential games since Pontryagin's time [90]. Whereas dynamic programming, ADP, and other local value function approximations reviewed in previous sections are closed-loop solutions to the differential game, the PMP gave open-loop admissible controls. Closed-loop strategies imply that the strategy is a function of both time and state, which changes as the system evolves, based on the current state of

the system. In an open-loop solution setting, both players formulate their strategy at the moment the system starts to evolve, based on information available at the time: the system dynamics, the objective function, and initial conditions. This strategy is only a function of time, and cannot be changed once the system evolves. The saddle-point solution is the combination of strategies of both players which are secured against any attempt by one player to unilaterally change his strategy (the definition of a Nash equilibrium).

In the context of robust control under the $\mathcal{H}_\infty$ and minimax formulations, a representative work is Boltyansky [91] where a version of the PMP was developed for the minimax control of systems with unknown parameters from a given finite set. This work was generalized by Boltyanski [92] to include problems with Bolza and Lagrange forms of objective functions. A recent book by Boltyanski [93] summarized the developments in this area.

Algorithms based on the PMP are often referred to as *indirect trajectory optimization*, owning to the fact that an extra step of forming the necessary conditions is done first before a discrete optimization problem is obtained. In contrast, *direct trajectory optimization* transcribes a continuous-time optimal control problem directly into an equivalent *nonlinear programming problem (NLP)* [94]. This is done by parameterizing the state and control spaces using a parametric representation and solving the resulting optimization by satisfying the constraints at particular *collocation points*. This is an idea borrowed from the numerical solution of differential equations [95], for which the collocation method is a well-established approximate method. The use of collocation in optimal control is a a NLP problem, which can in turn, be converted to a *sequential quadratic programming (SQP)* problem [96] and solved with a numerical optimization solver such as SNOPT [97] to produce the locally optimal solutions.

One of the most popular direct trajectory optimization tools is pseudospectral methods [98]. It is a class of *direct collocation* methods where the state and control vectors are parametrized by Lagrange polynomials [99] and collocating the differential-algebraic

equations using nodes obtained from a Gaussian quadrature. Various strategies have been proposed for the form of the collocation points, the three most commonly used are the *Legendre-Gauss(LG)*, *Legendre-Gauss-Radau (LGR)*, and *Legendre-Gauss-Lobatto (LGL)* points, which are obtained from the roots of a Legendre polynomial, belonging to a more general class of Jacobi polynomials [100]. Orthogonal polynomial enjoy the property of spectral accuracy [101], which states that the approximation enjoys an exponential convergence as the degree of the polynomial is increased, and hence avoid the so-called Runge's phenomenon [101]. Other forms of polynomial approximations, such as splines, RBFs, and wavelets, do not have spectral accuracy. In addition, as the derivatives of orthogonal polynomials can be written in terms of the same orthogonal polynomials, mechanical systems in terms of position, velocity, acceleration can be more compactly represented.

Comparing direct and indirect methods for trajectory optimization, it can be noted that both require initial guesses for the solutions to be in the neighborhood of the optimum in order to produce good results. Indirect methods tends to enjoy more accurate overall solution with assurances on its local optimality. But forming the analytical expressions for the necessary conditions of optimality can be infeasible for large scale nonlinear dynamics, where direct methods have an advantage. Direct methods can incorporate state and input constraints into the optimization framework, in addition to enjoying more readily available solvers as well as being more robust in convergence with respect to inaccurate initial guesses. A broad overview of numerical methods in optimal control can be found in Betts [94] and Subchan [102]

Another advantage of trajectory optimization based solutions to optimal control is the ability to deal with under-actuated systems, which has less number of control inputs compared to the total number of degrees of freedom for the system. Unlike analytical tools in nonlinear control design, an optimization-based approach can determine the proper allocation of control inputs to the system, inherently handling the under-actuation. An example of this can be found in [103].

While direct trajectory optimization has been successful in addressing optimal control problems, applying it to robust control is not straightforward and a naive implementation tends to result in an expensive optimization problem. To the best of our knowledge, no successful implementation of direct trajectory optimization has been demonstrated for robust control problems.

1.2.6 Receding horizon control

One outstanding issue in both optimal and robust control is how an open-loop solution to the optimization problem can be applied in a closed-loop setting. While open-loop solutions are admissible control inputs and obey the dynamics, they cannot compensate for responses of the system that differ from the predicted model. Doing so would require feedback, which is the basis for all closed-loop systems.

One popular approach in applying open-loop optimization is *receding horizon control (RHC)*, also known as *model predictive control (MPC)*. In a RHC setting, an optimization problem is solved and applied over a small *planning horizon*. As the response of the system diverges from the predicted model, a new optimization problem is solved and the process is repeated until convergence. The advantage of RHC compared to traditional control designs lies in its ability to iteratively apply simple controllers to solve more complex problems. Similar to how constrained optimal control problems can be tackled by iteratively solve a series of unconstrained ones [104], the receding-horizon formulation can be used to design closed-loop controls with open-loop trajectories. Franz [105] computed B-spline parameterized trajectories with a SQP-based method, and applied the trajectories in a RHC formulation. A survey by Diehl [106] reviewed related methods with Newton type optimization methods and SQP in the context of RHC for optimal control.

We refer readers to Kwon [107] for a review of general RHC methods in optimal control and instead focus the subsequent discussion on the use of RHC in robust control prob-

lems. The issue of robustness in RHC can be considered both passively and actively. A passive approach designs a RHC with nominal stability and consider stability margins for uncertainty and disturbances in retrospect. As shown in Magni [108], certain RHC designs are optimal in the sense that it is also optimal for a modified optimal control problem spanning over an infinite horizon. Therefore, the RHC is robust with respect to sector bounded input uncertainties, a property inherited as the sampling time goes to zero.

An active approach to robust RHC directly models uncertainty and disturbance in an $\mathcal{H}_\infty$ sense, and hence is more relevant to our discussion. It is, however, not a trivial endeavor since this formulation requires solving the HJI in some form. Chen [109], Blauwkamp [110], and Magni [111] [112] applied a game theoretic approach to nonlinear RHC, for which open-loop solutions to linearized $\mathcal{H}_\infty$ subproblems were obtained and applied in a receding-horizon fashion. While this simplified the designs considerably, in certain cases, no feasible solution can be found at all as one input signal must reject all possible disturbances. This is a limitation of the linear nature of the design. More recently, Yu [113] proposed a LMI-based solution to the HJI, on a linearized model of the original nonlinear system, which guaranteed $\mathcal{L}_2$-performance. While still linear in nature, it is an improvement on previous open-loop solutions. Related is Gautam [114], which presented a receding horizon style coordinated control framework involving dynamically decoupled subsystems which are required to reach a consensus condition, with the underlying $\mathcal{H}_\infty$ problem solved with LMIs.

Overall, robust RHC remains a largely unsolved problem. This is due to the fact that solving the HJI, even in a limited sense, is computationally burdensome and cannot be easily applied in an iterative setting.

1.2.7 Multi-model robust control

Given that nonlinear $\mathcal{H}_\infty$ designs in terms of the HJI are difficult to pursue, a heuristic approach to designing robust optimal feedback control called *multi-model control* evaluates the performance of a given control law over a distribution of possible models. A distribution of models allow the optimization to consider the expected cost over a range of possible outcomes and account for the maximum cost (or the worst case). In essence, the philosophy of multi-model control does not fundamentally differ from that of $\mathcal{H}_\infty$ control, though in practice, the problem structure differs significantly from traditional $\mathcal{H}_\infty$ designs.

The use of a distribution of models has been used in a number of different setting in robotics, machine learning, and state estimation and control. In reinforcement learning, for example, Cutler [115] proposed the use of multifidelity simulators, which effectively is a distribution of possible world models in which the learning is based on. In robotics, Cunningham [116] designed the controller such that it execute a policy from a set of plausible closed-loop policies, derived from models based on partially observable Markov decision processes. Ensemble control [117] is a class of methods which model a family of independent, structurally identical, finite-dimensional systems with variations in system parameters. A single common controller, which can be designed under an optimal control framework [118], steers the family of models between points of interests. Ensemble control has been applied in robotics settings [119], but is restrictive in the forms of structural parametric uncertainties that can be modeled.

In multi-model control designs, an early work by Varga [120] designed optimal output feedback control for multi-model linear-quadratic systems. More rigorously, Poznyak [121] formulated necessary conditions for multiple model LQ systems using the maximum principle, which was recently extended in [122] and Miranda [123]. For multi-model nonlinear systems, Azhmyakov [124] was the first to formulate necessary and sufficient conditions using dynamic programming. A recent book by Boltanski [93] summarized these approaches

from the prospective of both the dynamic programming principle as well as Pontryagin's minimum principle. Closely related is Whitman [125], where the implementation of a dynamic programming-based control design for the multi-model pendulum swing-up problem demonstrated the effectiveness of multi-model designs in addressing uncertainty. While these attempts are more expressive in modeling uncertainty and thus more robust compared to the traditional $\mathcal{H}_\infty$ approach, they are unattractive from a computational viewpoint: the two-point boundary value problem arising from the maximum principle is difficult to solve while the dynamic programming-based approaches are ultimately limited in the dimensionality of the system that they can handle.

Multi-model design are also used in trajectory optimization where only locally optimal solutions are desired. McNaughton [126] designed the system CASTRO using a direct collocation based trajectory optimizer DIRCOL and simultaneously optimized trajectories for multiple models, each using different estimates for the system parameters. The resulting system was demonstrated on a two-link pendulum swing up problem. More recently, Atkeson [127] gave a policy optimization approach where first and second order analytic gradients were used to obtain local solutions to multi-model dynamics. While these approaches are computationally attractive, they lack the rigorous theoretical guarantees for convergence and optimality. Depending on the specific baseline trajectory optimization method used, it can also be difficult to implement and parallelize the algorithms.

1.3 Formulation

1.3.1 System dynamics

In this section, we describe the general formulation of nonlinear robust control as an optimization problem that will be used throughout the remainder of the thesis. Consider a

nonlinear system

$$\begin{aligned} \dot{x} &= f(x) + g(x)u(t) \\ z &= [h(x), u]^T \\ x(t_0) &= x_0, \end{aligned} \tag{1.1}$$

with $x(t) \in \mathcal{X}$ is the *state* vector, $z(t) \in \mathcal{Z}$ is the *output* vector, $u(t) \in \mathcal{U}$ is the vector of *controlled inputs* to the system, x_0 is a vector of *initial states*, and the matrices $f(x)$, $g(x)$, and $h(x)$ are the *system matrices* that are smooth vector fields not explicitly parameterized by time. This form is commonly referred to as the *control affine form* since it linear in the actions but nonlinear with respect to the states. We assume, without a loss of generality, that the origin $x(t) = 0$ is an *equilibrium* of the system, i.e., $x(t) = 0, \forall t \in [t_0, \infty)$. Given any initial states x_0, we call $u(x, t)$ a *feedback control law* or *policy* that is an explicit function of the states. When u is not parameterized by the states, then we call $x(t, t_0, x_0, u)$ the *trajectory* of the system from initial time t_0 to final time t_f.

In robust control, we wish to consider potential sources of uncertainty that significantly affect the performance of the system. Typically, these uncertainties may include

- External disturbances: external perturbations to the system which are potentially time-varying and can alter the equilibrium state of the system
- Parametric uncertainty: parameters within the system dynamics, such as mass and stiffness, that are difficult to determine with complete accuracy or known to vary due to degradation and wear over time
- Unmodeled dynamics: higher-order dynamics (such as actuator dynamics) that has not been modeled but has has a significant effect on the response of the system in real life

Where in traditional model-based control designs, these uncertainties are assumed to be addressed by the feedback controls *implicitly*, we wish to *explicitly* design the controller to account for these uncertainties. We make the assumption that the disturbances are *bounded* and the parametric uncertainties are *structured*. These assumptions and the corresponding modifications to the system dynamics are stated below.

1.3.2 Uncertainty modeling

To model an external disturbance, we assume that the effect of the disturbance on the system dynamics is structurally *additive*. We alter the dynamics in (1.1) and add a disturbance term $w \in \mathcal{W}$ with an associated coefficient to form the system

$$\begin{aligned} \dot{x}(t) &= f(x) + g_1(x)w(t) + g_2(x)u(t) \\ z(t) &= [h(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{1.2}$$

where $f(x)$, $g_1(x)$ and $g_2(x)$ are the new system matrices. Consider the space $\mathcal{L}_2[0,\infty)$ for all piecewise-continuous inputs defined on $[0,\infty)$ satisfying

$$\int_0^\infty \|v(t)\|^2 dt < \infty. \tag{1.3}$$

The nonnegative number

$$\|v\|_2 \equiv \left(\int_0^\infty \|v(t)\|^2 dt \right)^{\frac{1}{2}} \tag{1.4}$$

is the $\mathcal{L}_2$-*norm* of v. Suppose the disturbance term w is a function in $\mathcal{L}_2[0,\infty)$. We use the concept of $\mathcal{L}_2$-*gain* and $\mathcal{H}_\infty$-*norm* to bound the response of the system under disturbance.

Definition 1.1. ($\mathcal{L}_2$-gain). Given an external disturbance $w \in \mathcal{W}$ and the corresponding output $z(t) \in \mathcal{Z}$, the input-output relationship

$$\int_{t_0}^{t_f} \|z(t)\|^2 \, dt \leq \gamma^2 \int_{t_0}^{t_f} \|w(t)\|^2 \, dt + \kappa(x_0), \quad \forall t_f > t_0, \tag{1.5}$$

is bounded for any $t_f > 0$, initial states x_0, a scalar γ, and some bounded function κ such that $\kappa(0) = 0$. We define the $\mathcal{L}_2$-*gain* from w to z to be the ratio between the $\mathcal{L}_2$-norm of the output and the $\mathcal{L}_2$-norm of the input, which is bounded by γ according to (1.5).

In other words, for any disturbance $w \in \mathcal{L}_2[0, \infty)$, the response of the system from the initial states x_0 is defined for all $t_f \geq 0$, which produces the output z which is also a function in $\mathcal{L}_2[0, \infty)$.

Definition 1.2. ($\mathcal{H}_\infty$-norm). The $\mathcal{H}_\infty$-*norm* of the system is the *maximum gain* of the system for all $\mathcal{L}_2$-bounded disturbances, that is, the $\mathcal{L}_2$-gain of the system from w to z is the induced-norm from $\mathcal{L}_2$ to $\mathcal{L}_2$:

$$\mathcal{H}_\infty = \sup_{0 \neq w \in \mathcal{L}_2(0,\infty)} \frac{\|z(t)\|_2}{\|w(t)\|_2}, \qquad x(t_0) = 0, \tag{1.6}$$

where for any $v : [t_0, t_f]$,

$$\|v\|^2_{2,[t_0,t_f]} \equiv \int_{t_0}^{t_f} \sum_{i=1}^{m} |v_i(t)|^2 \, dt. \tag{1.7}$$

The goal of *disturbance rejection* control is to render locally the $\mathcal{L}_2$-gain of the system to be less than or equal to γ such that all state trajectories are bounded and converge to the equilibrium point. Clearly, this can be alternatively stated as the minimization of the $\mathcal{H}_\infty$-norm, which is commonly referred to as $\mathcal{H}_\infty$-*control*. Later in this chapter, we will formally state an optimization framework for the $\mathcal{H}_\infty$-norm minimization problem.

To model parametric uncertainties in the system, we assume that the uncertainties are structurally *additive* as opposed to a more general *multiplicative* form. This is a

restrictive assumption done to preserve the control-affine form of the system and simplify derivations for the solutions. We will revisit this assumption in Chapter 5 and present an alternative formulation which relaxes it. For now, we form the system dynamics with additive uncertainties as

Definition 1.3. (Uncertain affine system dynamics). The dynamics of the system is given by a set of first-order differential equations

$$\begin{aligned} \dot{x}(t) &= \Big[f(x) + \Delta f(x,\theta,t)\Big] + g_1(x)w(t) + \Big[g_2(x) + \Delta g_2(x,\theta,t)\Big]u(t) \\ z(t) &= [h(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{1.8}$$

where $x(t) \in \mathcal{X}$ is the *state* vector, $z(t) \in \mathcal{Z}$ is the *output* vector, $u(t) \in \mathcal{U}$ is the vector of *controlled inputs* to the system, x_0 is a vector of *initial states*, and the matrices $f(x)$, $g(x)$, and $h(x)$ are the *system matrices* that are smooth vector fields not explicitly parameterized by time, along with a set of admissible uncertainties $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$.

Definition 1.4. (Admissible uncertainties). The parameter $\theta \in \Theta$ is the uncertain system parameter which belongs to the set

$$\Theta = \{\theta | 0 \leq \theta \leq \theta_u\} \tag{1.9}$$

that is bounded above by θ_u. Parametrized by θ, the functions $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$ are uncertain functions belonging to the set of *admissible uncertainties*, defined by

$$\begin{aligned} \Delta f(x,\theta,t) &= H_2(x)F(x,\theta,t)E_1(x) \\ \Delta g_2(x,\theta,t) &= g_2(x)F(x,\theta,t)E_2(x), \end{aligned} \tag{1.10}$$

for $\|F(x,\theta,t)\|^2 \leq \beta$, bounded by the constant β. Here, H_2, E_1, and E_2 are appropriate

state-dependent weight matrices for $F(x, \theta, t)$.

Intuitively, we assume that the uncertainties are bounded by a sphere. This is a generalization of the admissible uncertainties used in *guaranteed-cost control* [128][33], referred to as *structured* and *matched*. It is useful for modeling uncertainties in nonlinear systems since it accounts for both external disturbances and parametric uncertainties. In conjunction with the definition of the $\mathcal{H}_\infty$ norm, we can formulate an optimization problem to address both disturbances and uncertainties under this form.

1.3.3 Optimization

Given the system dynamics in (1.8), we seek a control input u which stabilizes the system under the assumed admissible uncertainties. To accomplish this, we formulate the following optimization framework.

Definition 1.5. (Minimax objective function). The optimization objective take the form

$$J(x, u, w, \theta) = \min_{u \in \mathcal{U}} \max_{\substack{w \in \mathcal{W} \\ \theta \in \Theta}} \frac{1}{2} \int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2 \|w(t)\|^2 \right] dt, \tag{1.11}$$

given a scalar γ and a time horizon $t_f > t_0$. We assume that $x(t) = 0$ is an unique equilibrium point of the system with $u(t) = 0$ and $w(t) = 0$.

This optimization framework is called *minimax* in the sense that we try to minimize the objective function under the worst possible disturbances or parameter variations, which maximizes the same objective function. It is also clear that (1.11) can be interpreted as a *noncooperative, two-player, zero-sum differential game* where one player seeks to minimize

$$J_1(x, u, w, \theta) = \min_{u \in \mathcal{U}} \frac{1}{2} \int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2 \|w(t)\|^2 \right] dt, \tag{1.12}$$

while the other player seeks to maximize

$$J_2(x,u,w,\theta) = \max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2\|w(t)\|^2\right] dt. \tag{1.13}$$

These opposing objectives are related by

$$J_1(x,u,w,\theta) = -J_2(x,u,w,\theta). \tag{1.14}$$

This adversarial optimization problem is a *differential game*, since both players are constrained by the system dynamics, a set of ordinary differential equations, as well as the associated uncertainties. Furthermore, the game is *non-cooperative* as each player chooses a feasible strategy independent of the other player's strategy and the payoff depends on both players' strategies. The outcome is *zero-sum*, since one player's gain is exactly balanced by the other player's loss.

In the context of game theory, the equilibrium strategy for this differential game constitutes a *Nash equilibrium* [17], also called a *saddle-point*, which can be intuitively explained as a pair of strategies secured against any attempt by one player to unilaterally change his strategy. In a continuous-time setting, a pair of strategies (u^*, w^*) forms a Nash equilibrium if

$$J_1(x,u^*,w^*,\theta) \le J_1(x,u,w^*,\theta), \quad \forall u\in\mathcal{U}, \theta\in\Theta \tag{1.15}$$

$$J_2(x,u^*,w^*,\theta) \ge J_2(x,u^*,w,\theta), \quad \forall w\in\mathcal{W}, \theta\in\Theta. \tag{1.16}$$

Equivalently, the *saddle-point condition* indicates that

$$J(x,u^*,w,\theta) \le J(x,u^*,w^*,\theta) \le J(u,w^*,\theta), \quad \forall u\in\mathcal{U}, w\in\mathcal{W}, \theta\in\Theta. \tag{1.17}$$

Unlike static games, differential game theory utilizes the concept of both open and closed-

loop solutions. In an open-loop setting, both players formulate their strategy at the moment the system starts to evolve, based on information available at the time: the system dynamics, the objective function, and initial conditions. This strategy cannot be changed once the system evolves and the saddle-point solution is the combination of strategies of both players which are secured against any attempt by one player to unilaterally change his strategy. Closed-loop strategies imply that the strategy can be changed as the system evolves, based on the current state of the system. Mathematically, an open-loop strategy is the pair $(u(t), w(t), t_0)$ as functions of time only, and a closed-loop strategy is the pair $(u(x,t), w(x,t))$ as functions of both time and state. We will discuss open-loop solutions in detail in Chapter 4.3.

One key assumption to a differential-game formulation to robust control is the notation of *rationality*. Both players in this adversarial optimization problem are assumed to act rationally and independently. If this assumption is violated, then the robust controller will exhibit sub-optimality or conservatism. An extensive discussion of this issue can be found in Section 6.5, in the context of experimental results.

1.4 Thesis contributions

In this thesis, we aim to improve feedback control design by explicitly modeling sources of uncertainties and provide a tractable computational approach to robust control. We make three novel contributions to nonlinear robust control for the problem described in Section 1.3. Specifically,

In Chapter 3, we design global methods that provide exact and approximated solutions to the nonlinear robust control problem. We derive necessary and sufficient conditions for optimality in terms of a Hamilton-Jacobi-Isaacs equation, which forms the basis of a two-play differential game. A value function is given as the solution of the HJI, from which

optimal feedback control laws can be derived for both players. We interpret these feedback control laws, in the context of strategies for opposing players and the value function. For practical solutions of the HJI, we present a value function approximation approach, which relies on Fourier basis functions in connection with Galerkin's method.

In Chapter 4, we extend direct trajectory optimization to nonlinear robust control, where open-loop trajectories are obtained in accordance with necessary conditions for optimality. The proposed framework transforms a minimax optimization problem into a minimization problem with complementarity conditions. The resulting problem is solved by trajectory optimization, which transcribes a continuous-time robust control problem into an equivalent discrete form by parameterizing the state and control spaces using global polynomials and collocating the differential-algebraic equations using nodes obtained from a Gaussian quadrature. We apply the open-loop trajectories in a receding-horizon control setting.

In Chapter 5, we develop a heuristic approach to robust control design based on the concept of multiple models, which evaluates the performance of a given control law over a distribution of possible system dynamics. A distribution of models allow the optimization to consider the expected cost over a range of possible outcomes and account for the maximum cost for the worst-case scenario. We improve on the approaches presented in Chapter 3 and Chapter 4 by removing restrictive assumptions made on the forms of uncertainties, which enable the approach to compensate for effects such as unmodeled dynamics. We develop these results in the context of necessary and sufficient conditions for individual value functions similar to Chapter 3 and utilize the robust trajectory optimization approach in Chapter 4 to compute open-loop trajectories that simultaneously satisfied the optimality conditions for multiple models. We demonstrate the use of these trajectories in a receding-horizon setting.

These contributions are united with the framework of spectral approximation, which we

introduce in Chapter 2. Through the use of a least-squares approximation framework, we introduce polynomial approximation which generates polynomials that exhibit properties of stability and convergence, in the form of a special class of polynomials known as orthogonal polynomials. In each chapter, we use a two-dimensional uncertain system as a benchmark problem to evaluate the proposed approach. In Chapter 6, we give simulation results for the three approaches on large scale systems and address the scalability of the proposed approaches in terms of dimensionality.

Compared to existing literature in robust control, this work is distinct in several aspects. First, we remove the restrictive assumptions on the forms of uncertainties that can be modeled and provide a general framework from which parametric system uncertainties, external disturbances, and unmodeled dynamics can be simultaneously addressed in a consistent optimization framework. Second, we provide a spectral approximation framework from which both global and local solutions can be obtained. Finally, we give computational techniques that significantly improve the scalability nonlinear robust control designs in comparison to previous work and provide solutions to high-dimensional problems that were previously intractable.

2

Background: Spectral Approximations

2.1 Motivation

A central idea in dynamic programming, reinforcement learning, and trajectory optimization is *function approximation*, the use of a combination of basis functions from a well-defined class that closely matches a given continuous function. In this chapter, we review a class of techniques in approximation theory and numerical analysis known as *spectral function approximation*. We do so with a focus on approximations in a *least-squares sense*, which generates polynomials that exhibit properties of *stability* and *convergence*, in the form of a special class of polynomials known as *orthogonal polynomials*. These function approximation techniques will be used in the subsequent chapters, in both single-dimensional (trajectory) and multi-dimensional (value function) settings. For a comprehensive overview of function approximation using spectral methods, we refer the readers to Askey [100], Quarteroni [101], Boyd [129], and Shen [130].

It is well known that any continuous function $f(x)$ can be represented in terms of n basis functions

$$\{\phi_0(x), \phi_1(x), \ldots, \phi_n(x)\} \tag{2.1}$$

and a finite-dimensional approximation of $f(x)$ can be done using a linear combination of

these basis functions in the form

$$\tilde{f}(x) = \sum_{i=1}^{n} c_i \phi_i(x), \tag{2.2}$$

where the variables c_i are coefficients associated with the basis functions $\phi_i(x)$. In this form, the monomial basis

$$\phi_k(x) = x^k \tag{2.3}$$

gives a polynomial approximation of $f(x)$. The Weierstrass approximation theorem states that for every continuous function defined on an internal $[a, b]$, there exists a polynomial $P_n(x)$ of degree n such that

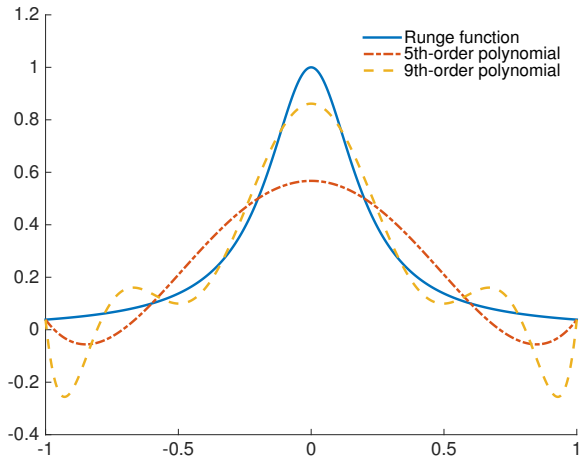
$$\lim_{n\to\infty} \left(\max_{a\le x\le b} |f(x) - P_n(x)| \right) = 0. \tag{2.4}$$

In other words, $f(x)$ can be uniformly approximated by $P_n(x)$ to arbitrary accuracy when n is sufficiently large. While the Weierstrass theorem opens up the possibility of using polynomials for the purposes of approximation, it does not specify the form of the polynomial. In particular, it does not specify constraints on the basis function or locations at which the original function $f(x)$ is to be sampled. As it turns out, certain choices of basis functions and sampling schemes can lead to instability. As demonstrated by the so-called *Runge's phenomenon*, when a function is sampled and interpolated at *equidistant nodes*

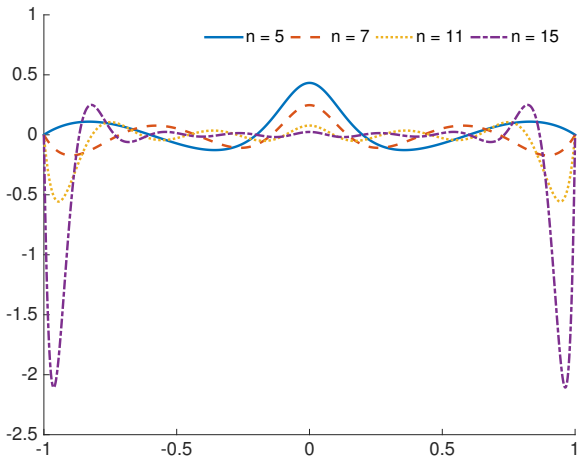
$$x_i = \frac{2i}{n} - 1, \quad i \in \{0, 1, \ldots, n\}, \tag{2.5}$$

the error of the approximation

$$\lim_{n\to\infty} \left(\max_{-1\le x\le 1} |f(x) - P_n(x)| \right) = \infty \tag{2.6}$$



(a) Blue: Runge function. Green: 5th-order polynomial approximation. Red: 9th-order polynomial approximation.



(b) Approximation error for 5th, 7th, 11th, and 15th-order polynomial approximation using equidistant nodes.

Figure 2.1: Runge's phenomenon for the function $f(x) = \frac{1}{1+25x^2}$.

grows without bound as the degree n of the polynomial increases. Fig. 2.1(a) demonstrates Runge's phenomenon for the function

$$f(x) = \frac{1}{1 + 25x^2}, \tag{2.7}$$

where 5th and 9th-order polynomial approximations are constructed using equidistant nodes. While a higher-order approximation is more accurate in the center, the errors are higher in the boundaries compared to a lower-order approximation. Fig. 2.1(b) shows the approximation error for orders $n = 5, 7, 11, 15$ where the errors continue to accumulate despite increasing the approximation order.

Runge's phenomenon shows that when attempting to approximate a given function, the selection of nodes and the choice of basis functions play important roles in the quality of the approximation. A *stable* approximation indicates that the approximation does not

exhibit Runge's phenomenon, while the *rate of convergence* indicates the rate in which the approximation error is decreased when the degree of the polynomial is increased.

2.2 Least-squares approximation

We pursue a well-formed polynomial approximation in the *least-squares sense*, as typically, there are more data (interpolation nodes) than the unknowns (coefficients of basis functions), creating an *overdetermined* system. In this formulation, the coefficients are obtained by evaluating a candidate polynomial and minimizing the sum of the squares of the errors made at the interpolation nodes.

Consider the *weighted* $\mathcal{L}_2$-norm of a function $f(x)$

$$\|f\|_{2,w} = \left(\int_a^b (f(x))^2 w(x)\, dx \right)^{\frac{1}{2}}. \tag{2.8}$$

The least-squares approximation problem is formulated to determine the polynomial that is closest to $f(x)$ such that

$$\|f(x) - \tilde{f}^*(x)\|_{2,w} = \min_{\tilde{f}(x)} \|f(x) - \tilde{f}(x)\|_{2,w}. \tag{2.9}$$

In other words, the weighted $\mathcal{L}_2$-norm of the approximation errors are minimized. In terms of the basis functions in (2.2), this is the problem of determining coefficients c such that

$$c^* = \arg\min_c \|f(x) - \tilde{f}(x)\|_{2,w}, \tag{2.10}$$

which gives the orthogonal projection of $f(x)$ onto the finite-dimensional basis. Instead of minimizing the the $\mathcal{L}_2$-norm of the difference, we can minimize its square. Let E denote

the square of the weighted $\mathcal{L}_2$-distance between $f(x)$ and $\tilde{f}(x)$

$$E(c_0, \ldots, c_n) = \int_a^b w(x) \left(f(x) - \tilde{f}(x) \right)^2 dx, \tag{2.11}$$

which is a quadratic function of the coefficients. A *necessary condition* for optimality for this unconstrained minimization problem is

$$\left. \frac{\partial E}{\partial c_j} \right|_{c=c^*} = 0. \tag{2.12}$$

By taking the partial derivative and simplifying, we obtain

$$-2 \int_a^b w(x)\phi_j(x)f(x)\,dx + 2 \sum_{i=0}^{n} c_i^* \int_a^b w(x)\phi_i(x)\phi_j(x)\,dx = 0, \tag{2.13}$$

which implies that

$$\sum_{i=0}^{n} c_i^* \int_a^b w(x)\phi_i(x)\phi_j(x)\,dx = \int_a^b w(x)\phi_j(x)f(x)\,dx, \quad j = 0, \ldots, n. \tag{2.14}$$

It is clear that if the integral term on the left-hand side of (2.14) evaluates to a scalar, then the problem is simplified considerably and we can obtain a closed-form expression for c^* in terms of other variables. In the simplest case, this can be done using basis functions $\phi_i(x)$ and $\phi_j(x)$ associated with *orthonormal polynomials*, defined on the inner product with respect to a weight $w(x)$

$$\int_a^b \phi_i(x)\phi_j(x)w(x)\,dx = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}, \tag{2.15}$$

where δ_{ij} is the Kronecker delta function. When the basis function satisfy this orthogonality property, the integral on the left-hand side of (2.14) evaluates to 1 and can be removed

entirely, resulting in the coefficients

$$c_i^* = \int_a^b \phi_i(x) f(x) w(x)\, dx, \quad i = 0, \ldots, n. \tag{2.16}$$

For more general problems, with polynomials that satisfy

$$\int_a^b \phi_i(x)\phi_j(x)w(x)\, dx = \delta_{ij} = \begin{cases} \int_a^b (\phi_i(x))^2 w(x)\, dx & i = j \\ 0 & i \neq j \end{cases}, \tag{2.17}$$

with $\int_a^b (\phi_i(x))^2 w(x)\, dx$ that does not necessarily evaluates to 1, the basis functions form *orthogonal polynomials*. Consequently, the solution to the least squares problem is given by the coefficients

$$c_i^* = \frac{\int_a^b \phi_i(x) f(x) w(x)\, dx}{\int_a^b (\phi_i(x))^2 w(x)\, dx}, \quad i = 0, \ldots, n. \tag{2.18}$$

In this formulation, to approximate a given function $f(x)$ involves selecting a basis function $\phi(x)$ and computing the coefficients c. In the subsequent sections, we will describe different choices for orthogonal polynomials and a way of computing the (2.18) using Gaussian quadrature.

2.3 Orthogonal polynomials

Consider the weighted Sobolev space

$$\mathcal{L}_w^2(a, b) = \left\{ f : \int_a^b f^2(x) w(x)\, dx < +\infty \right\} \tag{2.19}$$

defined on an interval (a, b) with the weight function $w(x)$. An inner product for functions f and g defined on this space takes the form

$$(f, g)_w = \int_a^b f(x)g(x)w(x)\,dx. \tag{2.20}$$

We note that the the $\mathcal{L}_2$-norm can be expressed using the inner product

$$\|f\|_{\mathcal{L}_w^2} = (f, f)_2^{\frac{1}{2}} \tag{2.21}$$

and

$$(f, g)_2 = 0 \tag{2.22}$$

if f and g are *orthogonal* to each other. A polynomial

$$p_n(x) = x^n + a_{n-1}^n x^{n-1} + \cdots + a_0^{(n)} \tag{2.23}$$

with degree n is orthogonal in $\mathcal{L}_w^2(a, b)$ if

$$(p_i, p_j)_w = 0, \qquad \text{for} \quad i \neq j. \tag{2.24}$$

For any given positive weight function w(x), there exists a unique set of orthogonal polynomials that can be constructed as a *three-term recurrence* relation

$$\begin{aligned} p_0 &= 1 \\ p_1 &= x - \alpha_1 \\ p_n &= (x - \alpha_{n+1})p_n - \beta_{n+1}p_{n-1}, \quad n \geq 1, \end{aligned} \tag{2.25}$$

where the coefficients α and β are given by

$$\begin{aligned}
\alpha_1 &= \int_a^b w(x)x\,dx \Big/ \int_a^b w(x)\,dx \\
\alpha_{n+1} &= \int_a^b xwp_n^2\,dx \Big/ \int_a^b wp_n^2\,dx \\
\beta_{n+1} &= \int_a^b xwp_np_{n-1}\,dx \Big/ \int_a^b wp_{n-1}^2\,dx.
\end{aligned} \tag{2.26}$$

All orthogonal polynomials can be constructed using these three-term recurrence relations, and among the most popular classical orthogonal polynomials is the *Jacobi family* of polynomials, defined with the weight

$$w(x) = (1-x)^\alpha(1+x)^\beta \quad \text{for} \quad \alpha,\beta > -1,\ (a,b) = (-1,1). \tag{2.27}$$

By definition, Jacobi polynomials are orthogonal since the choice of weight function produces

$$\int_{-1}^{1} p_n(x)p_m(x)(1-x)^\alpha(1+x)^\beta\,dx = 0, \quad \text{for} \quad n \neq m. \tag{2.28}$$

Legendre and *Chebyshev* polynomials belong to the Jacobi family and can be generated by varying the parameters α and *beta*. For $\alpha = \beta = 0$, we form the Legendre polynomials. For $\alpha = \beta = -\frac{1}{2}$, we get the Chebyshev polynomials. For now, we focus mainly on the Legendre polynomials.

On an interval $(a,b) = (-1,1)$ with $w(x) = 1$, the Legendre polynomial is defined as

$$\begin{aligned}
L_0(x) &= 1 \\
L_1(x) &= x \\
L_{n+1}(x) &= \frac{2n+1}{n+1}xL_n(x) - \frac{n}{n+1}L_{n-1}(x), \quad n \geq 1.
\end{aligned} \tag{2.29}$$

It is easy to check that Legendre polynomials are orthogonal in $[-1, 1]$ since

$$\int_{-1}^{1} L_i(x)L_j(x)\,dx = \frac{2}{2i+1}\delta_{ij}. \tag{2.30}$$

The Legendre polynomial can also be defined in terms of its derivatives L' such that

$$L_n(x) = \frac{1}{2n+1}(L'_{n+1}(x) - L'_{n-1}(x)), \quad n \geq 1, \tag{2.31}$$

where the first and second derivatives are given by

$$\begin{aligned} L'_n(x) &= \sum_{k=0}^{n-1}(2k+1)L_k(x), \quad k+n \text{ is odd} \\ L''_n(x) &= (k+\frac{1}{2})(n(n+1) - k(k+1))L_k(x). \end{aligned} \tag{2.32}$$

A sequence $\{L'_i(x)\}$ is mutually orthogonal with respect to $w(x) = 1 - x^2$ since

$$\int_{-1}^{1} L'_i(x)L'_j(x)(1-x^2)\,dx = \frac{2i(i+1)}{2i+1}\delta_{ij}. \tag{2.33}$$

By using Legendre polynomials, we have fixed the form of the basis functions in (2.2). What remains is the computation of corresponding coefficients in the form of (2.18), which we will describe in the next section.

2.4 Gaussian quadrature

Previously in (2.18), we have shown that the coefficients in a least-squares polynomial approximation are given by

$$c_i^* = \frac{\int_a^b \phi_i(x) f(x) w(x)\, dx}{\int_a^b (\phi_i(x))^2 w(x)\, dx}, \quad i = 0, \ldots, n. \tag{2.34}$$

With the choice of orthogonal polynomials, the weight function $w(x)$ and basis functions ϕ_i are fixed. The evaluation of expression now depends on the computation of definite integrals, subject to the selection of samples for the function $f(x)$. We approach the integration problem numerically and show that the choice of interpolation nodes can in fact be given in association with numerical quadratures.

Weighted quadratures are numerical integration schemes of the form

$$\int_{-1}^{1} f(x) w(x)\, dx \approx \sum_{i=0}^{n} A_i f(x_i) \tag{2.35}$$

over an the interval $[-1, 1]$ where $w(x) > 0$ is a weight function. Quadratures of this form applies generally to intervals $[a, b]$ by a simple linear transformation. Let

$$\lambda(t) = \frac{b-a}{2} t + \frac{a+b}{2}. \tag{2.36}$$

Then

$$\int_a^b f(x)\, dx = \frac{b-a}{2} \int_{-1}^{1} f(\lambda(t))\, dt \approx \frac{b-a}{2} \sum_{i=0}^{n} A_i f(\lambda(t_i)), \tag{2.37}$$

which implies that

$$\int_a^b f(x)\, dx = \frac{b-a}{2} \sum_{i=0}^{n} A_i f\left(\frac{b-a}{2} t_i + \frac{a+b}{d-c} \right). \tag{2.38}$$

For brevity, we will use the form

$$\int_a^b f(x)w(x)\,dx \approx \sum_{i=0}^{n} A_i f(x_i) \tag{2.39}$$

in the subsequent derivations and assume that the transformation has been done implicitly.

Numerical quadratures operate by sampling the given function $f(x)$ at selected nodes (sometimes referred to as *abscissas*) and quantifying the *order* of the quadrature by examining the highest degree polynomial in the integrand for which the quadrature is exact. For example, three-point and five-point centered difference formulas are exact when applied to second-degree and fourth-degree polynomials, respectively. If the choice of nodes $x_0, x_1, \ldots, x_N$ are made *a priori*, for example, with equidistant nodes, then resulting scheme (known as Newton-Cotes) has the coefficient

$$A_i = \int_a^b w(x)\frac{\prod_{i\neq j}(x - x_j)}{\prod_{i\neq j}(x_i - x_j)}\,dx, \tag{2.40}$$

which can be shown to be exact if the integrand is a polynomial of degree at most n.

In this approach, the choice of nodes is fixed with $n+1$ number of unknown coefficients A_i, which are explicitly dependent on the nodes. As observed by Gauss, if we can determine the location of the nodes (for example, in a non-equidistant fashion) in conjunction with coefficients, then we have twice as many parameters to choose from to maximize the accuracy of the quadrature. With $2n + 2$ degrees of freedom, the result is a higher degree of accuracy for the quadrature without needing to increase the number nodes, in a scheme known as *Gaussian integration*.

Consider the polynomial $f(x)$ of degree $2n + 1$ in the form

$$f(x) = q(x)p(x) + r(x), \tag{2.41}$$

where polynomials $p(x)$, $q(x)$, and $r(x)$ are all of degree n. $p(x)$ and $q(x)$ are orthogonal with respect to a weight function $w(x)$, i.e.,

$$\int_a^b p(x)q(x)w(x)\,dx = 0. \tag{2.42}$$

If $x_0, x_1, \ldots, x_n$ are the roots of $q(x)$, then

$$f(x_i) = r(x_i), \tag{2.43}$$

and the quadrature takes the form

$$\int_a^b f(x)w(x)\,dx = \int_a^b \left[q(x)p(x) + r(x)\right] w(x)\,dx. \tag{2.44}$$

Since $q(x)$ and $p(x)$ are assumed to be orthogonal, we can simplify the integral to be

$$\int_a^b f(x)w(x)\,dx = \int_a^b r(x)w(x)\,dx = \sum_{i=0}^{n} A_i r(x_i) = \sum_{i=0}^{n} A_i f(x_i). \tag{2.45}$$

In other words, we have demonstrated that by selecting the roots of an orthogonal polynomial, we can obtain a quadrature to accurately integrate functions of degrees at most $2n+1$. This Gaussian quadrature is optimal in the sense that it is impossible to pick $\{x_j, w_j\}_{i=0}^{n}$ for a quadrature that is accurate for $f(x)$ of degree $2n+2$.

All orthogonal polynomials can be shown to yield roots x_j that are *real*, *simple* (of multiplicity one), and are in the interval (a, b) for which the polynomial is defined on. In particular, since Legendre polynomials are solutions to Legendre's differential equation

$$\frac{d}{dx}\left[(1-x^2)\frac{d}{dx}P_n(x)\right] + n(n+1)P_n(x) = 0, \tag{2.46}$$

Picard's existence theorem guarantees that the all roots are distinct and real. This is a

useful property, as the use of Legendre polynomials with Gaussian nodes results in the *Gauss-Legendre quadrature*, with the weight function

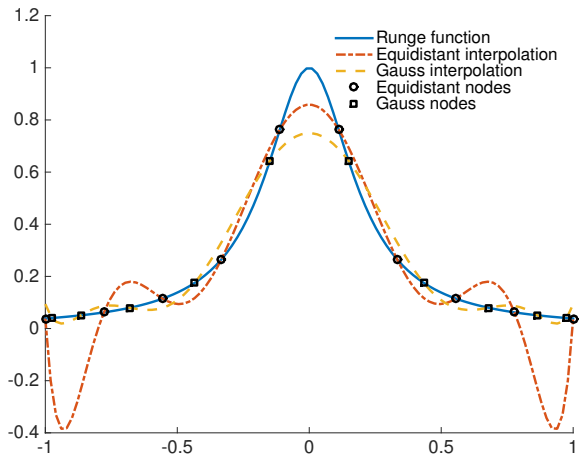
$$w_i = \frac{2}{(1 - x_i^2)\,[L_n'(x_i)]^2}. \tag{2.47}$$

The location of interpolation nodes are given by the roots of a n-degree Legendre polynomial. Table 2.1 gives node locations and weights for Legendre polynomials of degree at most $n = 5$.

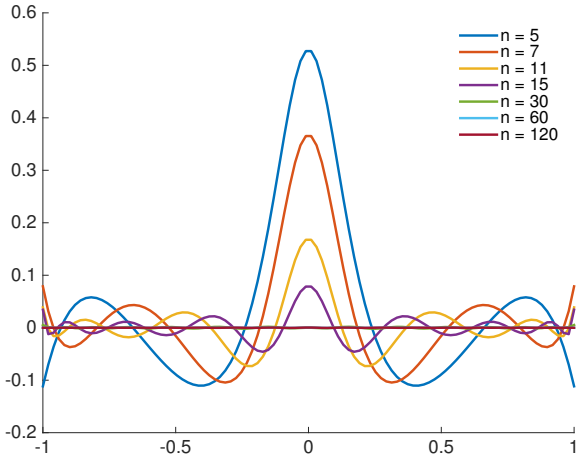
Table 2.1: Node location and weights for Gauss-Legendre quadratures

Number of nodes n	Node location x_i	Weights w_i
1	0	2
2	$\pm\sqrt{1/3}$	1
3	0 $\pm\sqrt{3/5}$	$\frac{8}{9}$ $\frac{5}{9}$
4	$\pm\sqrt{(3-2\sqrt{6/5})/7}$ $\pm\sqrt{(3+2\sqrt{6/5})/7}$	$\frac{(18+\sqrt{30})}{36}$ $\frac{(18-\sqrt{30})}{36}$
5	0 $\pm\frac{1}{3}\sqrt{(5-2\sqrt{10/7})}$ $\pm\frac{1}{3}\sqrt{(5+2\sqrt{10/7})}$	$\frac{128}{225}$ $\frac{332+12\sqrt{80}}{900}$ $\frac{332-12\sqrt{80}}{900}$

For the problem of approximating the Runge function (2.7), Fig. 2.2 compares Gauss nodes with traditional equidistant nodes. Fig. 2.2(a) shows the result of the two approximation schemes compared to the true Runge function, where square and circle markers indicates the location of the respective nodes. Fig. 2.2(b) plots the error of the Gauss nodes for polynomials of increasing complexity. Compared to the error plot in Fig. 2.1(b), we see that the choice of Gauss nodes results in a convergent approximation which avoids



(a) Blue: Runge function. Green: approximation with equidistant nodes. Red: approximation with Gauss nodes. Circle: location of equidistant nodes. Square: location of Gauss nodes.



(b) Approximation error for 5th, 7th, 11th, 15th, 30th, 60th, and 120th-order polynomial approximation using Gauss nodes.

Figure 2.2: Approximations for the function $f(x) = \frac{1}{1+25x^2}$.

Runge's phenomenon.

The quadratures we have discussed thus far do not include nodes at the end points ± 1 of the integration interval. For certain applications, such as boundary value problems discussed in Chapter 4, end points must be explicitly included to enforce Dirichlet boundary conditions. Simple variations on the Gauss-Legendre quadrature include the *Gauss-Radau* and *Gauss-Lobatto* quadratures. In Gauss-Radau, either (but only one) endpoint can be included as a node in the approximiation, which leaves $2n-1$ remaining parameters with an accuracy of $2n-2$. In Gauss-Lobatto, both endpoints are included as nodes, which results in an accuracy of $2n-3$. Gauss-Lobatto quadratures take the form

$$\int_{-1}^{1} f(x)w(x)\,dx \approx \frac{2}{n(n-1)}(f(1)+f(-1)) + \sum_{i=2}^{n-1} w_i f(x_i) + R_n, \tag{2.48}$$

where the weight w_i is given by

$$w_i = \frac{2}{n(n-1)(P_{n-1}(x_i))^2}, \quad x_i \neq \pm 1, \tag{2.49}$$

and the remainder R_n is

$$R_n = \frac{-n(n-1)^3 2^{2n-1}[(n-2)!]^4}{(2n-1)[(2n-2)!]^3} f^{2n-2}(\xi), \quad -1 < \xi < 1. \tag{2.50}$$

The interpolation nodes are given by the roots of $P'_{n-1}(x)$. Table 2.2 gives node locations and weights for Gauss-Lobatto quadratures of degree at most $n = 5$. In terms of con-

Table 2.2: Node location and weights for Gauss-Lobatto quadratures

Number of nodes n	Node location x_i	Weights w_i
3	0	$\frac{4}{3}$
	± 1	$\frac{1}{3}$
4	$\pm\sqrt{\frac{1}{5}}$	$\frac{5}{6}$
	± 1	$\frac{1}{6}$
3	0	$\frac{32}{45}$
	$\pm\sqrt{\frac{3}{7}}$	$\frac{49}{90}$
	± 1	$\frac{1}{10}$

vergence, there are no differences between the choice of Gauss, Gauss-Radau, and Gauss-Lobatto nodes. For general orthogonal polynomials, is rate of convergence is exponential in terms of the degrees of the approximation

$$\|f(x) - \tilde{f}^*(x)\|_{2,w} \sim e^{-n}. \tag{2.51}$$

In other words, for a sufficiently smooth function, as the degree n of the polynomial approximation $\tilde{f}^*$ is increased, there is an exponential decrease in the corresponding error.

This a property known as *spectral accuracy*, an unique property not enjoyed by other forms of function approximations, such as splines, RBfs, and wavelets [129].

2.5 Fourier approximation

For the approximation of periodic functions, the orthogonal polynomials described in the previous section is not appropriate since the polynomials are not periodic by definition. Instead, we describe the use of Fourier basis functions to achieve similar spectral approximation of periodic functions and we demonstrate that these trigonometric polynomials satisfies the orthogonality conditions similar to orthogonal polynomials.

Consider a periodic function $f(x)$ on the interval $[0, 2\pi]$. Periodicity indicates that $f(x)$ satisfies

$$f(x + 2\pi) = f(x). \tag{2.52}$$

We use $\mathcal{L}^2(0, 2\pi)$ to denote the space of complex-value functions that are square integrable over $[0, 2\pi]$, that is,

$$\mathcal{L}^2(0, 2\pi) = \left\{ f : (0, 2\pi) \to \mathcal{C} \quad \text{such that} \quad \int_0^{2\pi} |f(x)|^2 \, dx < \infty \right\}. \tag{2.53}$$

The inner product of functions f and g is defined as

$$(f, g) = \int_0^{2\pi} f(x)g(x) \, dx \tag{2.54}$$

and the norm is

$$\|f\|_{\mathcal{L}^2_{0,2\pi}} = \sqrt{(f, f)}. \tag{2.55}$$

Fourier trigonometric polynomials are defined using the basis

$$\phi_k(x) = e^{ikx} = \cos(kx) + i\sin(kx), \quad k = 0, \pm 1, \pm 2, \ldots \tag{2.56}$$

It is clear that the Fourier basis (2.56) is orthogonal since

$$(\phi_j, \phi_k) = \int_0^{2\pi} \phi_j(x)\phi_k(x)\,dx = \int_0^{2\pi} e^{i(j-k)x}\,dx = 2\pi\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}, \tag{2.57}$$

where δ_{ij} is the Kronecker delta function. The polynomial is then the linear combination of the basis in the form

$$F = \sum_{k=-\infty}^{\infty} f_k \phi_k, \tag{2.58}$$

where the coefficient is

$$f_k = \frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ikx}\,dx = \frac{1}{2\pi}(f, \phi_k). \tag{2.59}$$

For the choices

$$\begin{aligned} a_k &= \frac{1}{2\pi}\int_0^{2\pi} \left(\alpha(x)\cos(kx) + \beta(x)\sin(kx)\right)\,dx \\ b_k &= \frac{1}{2\pi}\int_0^{2\pi} \left(-\alpha(x)\cos(kx) + \beta(x)\sin(kx)\right)\,dx, \end{aligned} \tag{2.60}$$

the Fourier coefficients can be written as

$$f_k = a_k + ib_k, \quad k = 0, \pm 1, \pm 2, \ldots \tag{2.61}$$

It is also clear that for real valued functions,

$$f_{-k} = f_k \quad \forall k. \tag{2.62}$$

Following the least squares function approximation described in (2.9), we have

$$\|f(x) - \tilde{f}_N^*(x)\|_{\mathcal{L}^2(0,2\pi)} = \min_{\tilde{f}(x)} \|f(x) - \tilde{f}(x)\|_{\mathcal{L}^2(0,2\pi)}, \tag{2.63}$$

where $f_N^*(x)$ is a finite truncation of order N, which gives the approximation

$$f(x) \approx \sum_{k=-\frac{N-1}{2}}^{\frac{N-1}{2}} f_k^{(N)} e^{ikx}. \tag{2.64}$$

The coefficients are given by

$$f_k^{(N)} = \frac{1}{N} \sum_{j=0}^{N-1} f_j \exp\left(-\frac{2\pi ijk}{N}\right). \tag{2.65}$$

Similar to orthogonal polynomials, the spectral accuracy with respect to the number of nodes N is

$$\|f(x) - \tilde{f}(x)\|_{\mathcal{L}^2(0,2\pi)} \sim e^{-N} \tag{2.66}$$

which is an exponential convergence.

2.6 Summary

In this chapter, we presented a consistent framework in which continuous-time functions can be approximated as linear combinations of basis functions. This approximation is done in the least-squares sense, which evaluates a candidate polynomial and minimizes the sum of the squares of the errors made at the interpolation nodes. We've determined that by using orthogonal polynomials as basis functions and sampling at Gauss nodes, the resulting approximation avoids Runge's phenomenon and exhibits spectral accuracy.

The function approximation framework we have presented is done in the context of interpolation, where a known function $f(x)$ is given (as least in terms of samples). In the context of control problems, the function that we wish to approximate is the result of a constrained optimization problem and is unknown. While we can still select the form of basis functions based on the orthogonality conditions, the coefficients for the basis function are computed through optimization instead of quadratures. This is done by enforcing system dynamics and associated constraints at *collocation points*, which are interpolation nodes described in Section 2.4. We will address these issues in detail in the subsequent chapters.

3

Global Solutions

3.1 Motivation

In Section 1.3, we described a $\mathcal{H}_\infty$ robust control problem as a two-player differential game with parametric uncertainties in the system dynamics. The two players formulate respective strategies in a non-cooperative and zero-sum fashion, which results in a *Nash equilibrium* as a saddle-point strategy. In this chapter, we establish necessary and sufficient conditions for optimality and formulate global solutions to this differential game.

Global solutions are difficult to obtain in nonlinear optimization problems in general. In optimal control, the application of the dynamic programming principle in a continuous-time problem with nonlinear dynamics results in the Hamilton-Jacobi-Bellman (HJB) equation, a nonlinear first-order partial differential equation that is the necessary and sufficient condition for global optimality. For nonlinear robust control problems in the $\mathcal{H}_\infty$-style, the dual to the HJB is the Hamilton-Jacobi-Isaacs (HJI) equation, for which the solution is the Bellman value function. As reviewed in Section 1.2.2, these partial differential equations are extremely difficult to solve.

In this chapter, we characterize global solutions to the differential game formulation laid out in Section 1.3 from a value function approximation perspective. We reviewed related work in Section 1.2.3 and Section 1.2.4. We derive necessary and sufficient conditions for optimality for the $\mathcal{H}_\infty$ problem in Section 1.3 in terms of the Isaacs equation, then

approximate the value function as a linear combination of basis functions. We build upon the spectral approximation framework in Chapter 2, which suggests basis functions in terms of orthogonal polynomials, resulting in an approximation that exhibits spectral accuracy.

The result of this chapter is an characterization of the value function from the perspective of differential games, which is critical in obtaining explicit feedback control laws that guarantee the asymptotic stability of the closed-loop system, without requiring the explicit form of the parametric uncertainties. For the practical implementation of the proposed approach, we demonstrate the value function approximation framework for a simple two-dimensional differential game, which will be used as a baseline comparison for the methods in the subsequent chapters.

3.2 Necessary and sufficient conditions

In Section 1.3, we formulated a robust $\mathcal{H}_\infty$-control problem with the nonlinear dynamics

$$\begin{aligned} \dot{x}(t) &= \Big[f(x) + \Delta f(x,\theta,t)\Big] + g_1(x)w(t) + \Big[g_2(x) + \Delta g_2(x,\theta,t)\Big]u(t) \\ z(t) &= [h(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{3.1}$$

with $x(t) \in \mathcal{X}$ as the *state* vector, $z(t) \in \mathcal{Z}$ is the *output* vector, $u(t) \in \mathcal{U}$ and $w(t) \in \mathcal{W}$ are the minimizing and maximizing players, respectively, x_0 is a vector of *initial states*, and the matrices $f(x)$, $g(x)$, and $h(x)$ are the *system matrices* that are smooth vector fields not explicitly parameterized by time. The parameter $\theta \in \Theta$ is the uncertain system parameter which belongs to the set

$$\Theta = \{\theta | 0 \leq \theta \leq \theta_u\} \tag{3.2}$$

that is bounded above by θ_u. Parametrized by θ, the functions $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$ are uncertain functions belonging to the set of *admissible uncertainties*, defined by

$$\begin{aligned}\Delta f(x,\theta,t) &= H_2(x)F(x,\theta,t)E_1(x)\\ \Delta g_2(x,\theta,t) &= g_2(x)F(x,\theta,t)E_2(x),\end{aligned} \tag{3.3}$$

for $\|F(x,\theta,t)\|^2 \leq \beta$, which intuitively, assumes that the uncertainties are bounded by a sphere. Here, H_2, E_1, and E_2 are appropriate state-dependent weight matrices for $F(x,\theta,t)$. The minimax objective

$$J(x,u,w,\theta) = \min_{u\in\mathcal{U}} \max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2\|w(t)\|^2\right] dt \tag{3.4}$$

entails a two-player, noncooperative, zero-sum differential game over a horizon $t_f > t_0$. We begin the derivation of the necessary and sufficient condition for optimality for this differential game by characterizing its saddle-point solution. This is the *Bellman value function*

Definition 3.1. (Bellman value function). The value function $V(x,t)$ defines the cost-to-go from any initial state x and any time t such that

$$V(x,t) = \inf_{u\in\mathcal{U}} \sup_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(\tau)\|^2 - \gamma^2\|w(\tau)\|^2\right] d\tau. \tag{3.5}$$

In other words, the value function bounds the cost and provides the optimal cost-to-go for any state $x(t)$, i.e.,

$$V(x(t_f),t_f) \leq J^*(x,u^*,w^*,\theta) \leq V(x(t_0),t_0), \tag{3.6}$$

where $u^*(t)$ and $w^*(t)$ are the optimal controls of the system ($w^*(t)$ is optimal in the sense

that it provides the worst-case disturbance to the system). We use the notation

$$V_x(x,t) = \frac{\partial V(x,t)}{\partial x(t)}, \quad V_t(x,t) = \frac{\partial V(x,t)}{\partial t}, \tag{3.7}$$

to denote the row vectors of first partial derivatives of the value function with respect to t and $x(t)$, respectively. For brevity, we drop the time index t when it is immaterial.

We use basic optimization theory to derive the optimal control and the associated necessary and sufficient condition for optimality. First, we convert the constrained optimization problem in (3.4) to an unconstrained form by using the *Hamiltonian*.

Definition 3.2. (Hamiltonian). The Hamiltonian $H(x,\lambda,u,w,\theta)$ for the system dynamics in (3.1) and objective function (3.4) is

$$\begin{aligned} H(x,V_x,u,w,\theta) = V_x(x)\Big(&f(x) + \Delta f(x,\theta,t) + g_1(x)w(t) \\ &+ \left[g_2(x) + \Delta g_2(x,\theta,t)\right]u(t)\Big) + \frac{1}{2}\|z(t)\|^2 - \frac{1}{2}\gamma^2\|w(t)\|^2, \end{aligned} \tag{3.8}$$

where the λ is the *costate* or *adjoint* vector, which in this case, is the spacial derivative of the value function V_x.

The Hamiltonian combines the right-hand side of the system dynamics in (3.1) and the one-step cost within the integral in the objective function (3.4), therefore transforming a constrained optimization problem into an unconstrained one. In this unconstrained form, a necessary condition for optimality is *Isaacs' condition*, which states that

$$\inf_{u\in\mathcal{U}} \sup_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} H(x,V_x,u,w,\theta) = \sup_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \inf_{u\in\mathcal{U}} H(x,V_x,u,w,\theta). \tag{3.9}$$

Equivalently, this is

$$H(x,V_x,u^*,w,\theta) \le H(x,V_x,u^*,w^*,\theta) \le H(x,V_x,u,w^*,\theta). \tag{3.10}$$

For the Hamiltonian to be minimized over $\mathcal{U}$, the set of all possible controls, while maximized over $\mathcal{W}$, the set of all possible disturbances, we write

$$\min_{u\in\mathcal{U}} \sup_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} H(x, V_x, u, w, \theta) \leq 0. \tag{3.11}$$

Effectively, since we used $V_x(x)$ as the costate, the value function is the solution to this minimization problem. Now, we derive several simple identities which will be helpful in subsequent discussion. First,

$$\left\| H_2^T(x)V_x^T(x) - F(x,\theta,t)E_1(x) \right\|^2 \geq 0, \tag{3.12}$$

since by definition, a squared-norm is positive semi-definite. Also by definition, $\|F(x,\theta,t)\|^2 \leq \beta$, and hence we can expand the left-hand side as

$$\begin{aligned}\left\| H_2^T(x)V_x^T(x) - F(x,\theta,t)E_1(x) \right\|^2 &= V_x(x)H_2(x)H_2^T(x)V_x^T(x) \\ &\quad - 2V_x(x)H_2(x)F(x,\theta,t)E_1(x) + \beta^2 E_1^T(x)E_1(x) \geq 0.\end{aligned} \tag{3.13}$$

Rearranging the terms, we arrive at

$$V_x(x)H_2(x)F(x,\theta,t)E_1(x) \leq \frac{1}{2}\Big[V_x(x)H_2(x)H_2^T(x)V_x^T(x) + \beta^2 E_1^T(x)E_1(x)\Big]. \tag{3.14}$$

Likewise, we can establish that

$$\left\| I - \frac{1}{2}F(x,\theta,t)E_2(x) \right\|^2 = I - F(x,\theta,t)E_2(x) + \frac{1}{4}\beta^2 E_2^T(x)E_2(x) \geq 0, \tag{3.15}$$

where I is the identity matrix. This implies

$$F(x,\theta,t)E_2(x) \leq I + \frac{1}{4}\beta^2 E_2^T(x)E_2(x). \tag{3.16}$$

Going back to (3.11), we resolve the inner maximization problem first by taking the supremum of the Hamiltonian. This is accomplished by taking the Hamiltonian in (3.8) and substituting the definitions of the parametric uncertainties in (3.3), which forms an inequality

$$\begin{aligned}\sup_{w\in\mathcal{W},\theta\in\Theta} \quad H(x,V_x^T,u,w,\theta) \;\le\; V_x(x)\Big[\Big(f(x)+H_2(x)F(x,\theta,t)E_1\Big)+g_1(x)w(t)\\ +\Big(g_2(x)+g_2(x)F(x,\theta,t)E_2\Big)u(t)\Big]+\frac{1}{2}\|z(t)\|^2-\frac{1}{2}\gamma^2\|w(t)\|^2. \quad (3.17)\end{aligned}$$

Expanding the expression and applying the identities (3.14) and (3.16), we have

$$\begin{aligned}\sup_{w\in\mathcal{W}} H(x,V_x^T,u,w,\theta) \;\le\; V_x(x)f(x)+\Big[\frac{1}{2}V_x(x)H_2(x)H_2^T(x)V_x^T(x)+\frac{1}{2}\beta^2E_1^T(x)E_1(x)\Big]\\ +V_x(x)g_1(x)w(t)+\Big[V_x(x)g_2(x)\left(2I+\frac{1}{4}\beta^2E_2^T(x)E_2(x)\right)u(t)\Big]\\ +\Big(\frac{1}{2}h^T(x)h(x)+u^T(t)u(t)-\frac{1}{2}\gamma^2\|w(t)\|^2\Big), \quad (3.18)\end{aligned}$$

where the expressions for $F(x,\theta,t)$ have been eliminated in the Hamiltonian, and hence the supremum is no longer subject to $\theta\in\Theta$. According to the first-order necessary condition for optimality, we can obtain expressions for $u(t)$ and $w(t)$ by differentiating the Hamiltonian with respect to $u(t)$ and $w(t)$, respectively, then solving for the unknowns as functions of $V_x(x)$. Since most of the terms in (3.18) are not dependent on either $u(t)$ or $w(t)$, we can easily simplify the expressions to

$$\begin{aligned}u^*(t)&=-\left(2I+\frac{1}{4}\beta^2E_2^T(x)E_2(x)\right)^T g_2^TV_x^T(x)\\ w^*(t)&=\frac{1}{\gamma^2}g_1^T(x)V_x^T(x), \quad (3.19)\end{aligned}$$

which are the optimal feedback laws. Substituting these expressions back into (3.8), we

obtain the equation

$$V_x(x)f(x) + \frac{1}{2}V_x(x)\left[\frac{1}{\gamma^2}g_1(x)g_1^T(x) + H_2(x)H_2^T(x) - g_2(x)\left(2I + \frac{1}{4}\beta^2 E_2^T(x)E_2(x)\right)g_2^T(x)\right]V_x^T(x) + \frac{1}{2}h^T(x)h(x) + \frac{1}{2}\beta^2 E_1(x)E_1^T(x) \leq 0. \tag{3.20}$$

This is the *Hamilton-Jacobi-Isaacs equation* for the optimization problem laid out in (3.1). Given the admissible uncertainties in (3.3), this is a more general version of the Isaacs equation as compared to [33]. It can be made even more general by assuming non-affine dynamics, as described in [15], but as we shall see in Section 3.3, we can leverage the structures of this equation to improve computational efficiency.

By examining the form of the feedback control laws in (3.19), it is clear that while both players' strategies are non-cooperative and independent of each other, they both depend on the value function (or rather, the gradient of the value function). The sign difference in the feedback control laws arises from the zero-sum nature of the game, where both players are optimizing the same objective function, though in different directions. Where the minimizing player u follows a gradient descent of the value function, the maximizing player w follows a gradient ascent.

It is also worth noting that the feedback control laws obtained in (3.19) and the associated Isaacs equation (3.21) do not explicitly depend on the parametric uncertainty term $F(x, \theta, t)$. As we assumed that the squared-norm of $F(x, \theta, t)$ is bounded by a sphere, the uncertainty was taken into account in the formulation of the Isaacs equation that produced the value function as a solution, which accounts for all possible forms of $F(x, \theta, t)$ that satisfy the bounded norm assumption.

Theorem 3.3. (Necessary and sufficient condition for optimality). The Hamilton-Jacobi-

Isaacs equation

$$V_x(x)f(x) + \frac{1}{2}V_x(x)\left[\frac{1}{\gamma^2}g_1(x)g_1^T(x) + H_2(x)H_2^T(x) - g_2(x)\left(2I + \frac{1}{4}\beta^2 E_2^T(x)E_2(x)\right)g_2^T(x)\right]V_x^T(x) + \frac{1}{2}h^T(x)h(x) + \frac{1}{2}\beta^2 E_1(x)E_1^T(x) \leq 0. \tag{3.21}$$

is a necessary and sufficient condition for optimality for the optimization problem (3.4), subject to the system dynamics (3.1).

Proof. Since we derived the Isaacs equation using first-order necessary conditions on the Hamiltonian, it is clear that (3.21) is a necessary condition for (u^*, w^*) to be an unique optimum. To show that the Isaacs equation is also a sufficient condition for optimality, we use (3.18)

$$\begin{aligned}\sup_{w\in\mathcal{W}} H(x, V_x^T, u, w, \theta) \;\leq\; & V_x(x)f(x) + \left[\frac{1}{2}V_x(x)H_2(x)H_2^T(x)V_x^T(x) + \frac{1}{2}\beta^2 E_1^T(x)E_1(x)\right] \\ & + V_x(x)g_1(x)w(t) + \left[V_x(x)g_2(x)\left(2I + \frac{1}{4}\beta^2 E_2^T(x)E_2(x)\right)u(t)\right] \\ & + \left(\frac{1}{2}h^T(x)h(x) + u^T(t)u(t) - \frac{1}{2}\gamma^2\|w(t)\|^2\right),\end{aligned} \tag{3.22}$$

and note that the right hand side is the same as

$$\begin{aligned}H(x, V_x, u, w, \theta) =& V_x(x)f(x) + \frac{1}{2}V_x(x)\left[g_1(x)g_1^T(x) + H_2(x)H_2^T(x) - g_2(x)g_2^T(x)\right]V_x^T(x) \\ & + \frac{1}{2}h^T(x)h(x) + \frac{1}{2}\beta^2 E_1^T(x)E_1(x) - \frac{1}{2}\left\|w(t) - \frac{1}{\gamma^2}g_1^T(x)V_x^T(x)\right\|^2 \\ & - \frac{1}{2}V_x(x)g_2(x)\left[3I + \frac{1}{2}\beta^2 E_x^T(x)E_2(x)\right]g_2^T(x)V_x^T(x).\end{aligned} \tag{3.23}$$

It follows that since

$$V_x(x)f(x) + \frac{1}{2}V_x(x)\Big[g_1(x)g_1^T(x) + H_2(x)H_2^T(x) - g_2(x)g_2^T(x)\Big]V_x^T(x) + \frac{1}{2}h^T(x)h(x) + \frac{1}{2}\beta^2 E_1^T(x)E_1(x) \leq 0, \quad (3.24)$$

The first half of the expressions in the Hamiltonian reduces to zero and the rest simplifies to

$$-\frac{1}{2}\left\|w(t) - \frac{1}{\gamma^2}g_1^T(x)V_x^T(x)\right\|^2 - \frac{1}{2}V_x(x)g_2(x)\left[3I + \frac{1}{2}\beta^2 E_2^T(x)E_2(x)\right]g_2^T(x)V_x^T(x) \leq 0. \quad (3.25)$$

This indicates that (3.21) is also a sufficient condition for optimality. □

As is typical in optimal control formulations, the value function here is a natural Lyapunov function candidate. While it is trivial to verify the asymptotic stability of the closed loop system with no disturbance, let us consider the interpretation of the value function assuming a disturbance exists. By rearranging (3.6), we can obtain

$$V(x(t_f), t_f) - V(x(t_0), t_0) \leq \int_{t_0}^{t_f} \frac{1}{2}\Big[\gamma^2\|w(t)\|^2 - \|z(t)\|^2\Big]\, dt. \quad (3.26)$$

Taking the limit as as $t_f \to \infty$, the value function is zero at terminal time by definition. Splitting the integral and rearranging again, we obtain

$$\frac{1}{2}\int_{t_0}^{t_f} \|z(t)\|^2\, dt \leq \frac{1}{2}\int_{t_0}^{t_f} \gamma^2\|w(t)\|^2\, dt + V(x(t_0), t_0). \quad (3.27)$$

This is exactly the definition of the $\mathcal{L}_2$-gain defined in (1.5), indicating that the existence of the value function for this system implies that the $\mathcal{L}_2$-gain is bounded by γ.

3.3 Value function approximation

As discussed in Section 1.2.2, analytical solutions to the Isaacs equations cannot be expected for general nonlinear systems. Instead, we pursue the idea of *value function approximation*, reviewed in Section 1.2.3. Given the Isaacs equation in (3.21), the *Galerkin method* provides a convenient way of producing a discrete representation, where approximations to the value function can be carried out.

The Galerkin method is suitable for operator equations of the form

$$\mathcal{T}(V(x)) = l(x), \quad a \leq x \leq b, \tag{3.28}$$

where $\mathcal{T}$ is a linear operator, V is the unknown variable to be determined, and $l(x)$ is a function in terms of the independent variable x, bounded above and below by a and b, respectively. The boundary condition is defined as

$$S(x) = 0. \tag{3.29}$$

In the case of the Isaacs equation, which is a first-order nonlinear partial differential equation, $\mathcal{T}(V)$ is an operator involving the partial derivatives of value function $V(x)$ and $l(x)$ contain miscellaneous terms independent of the value function.

Following the approach presented in Chapter 2, we approximate the value function V as a weighted sum of a set of basis functions ϕ_i.

Definition 3.4. (Approximate value function). The approximate value function $\tilde{V}(x)$ is

$$\tilde{V}(x) = \sum_{i=1}^{n} w_i \phi_i(x), \tag{3.30}$$

where $\phi_i(x)$ is a vector of basis function with corresponding weights w_i.

We seek an approximation in the least-squares sense

$$\|V(x) - \tilde{V}^*(x)\|_{2,w} = \min_{\tilde{V}(x)} \|V(x) - \tilde{V}(x)\|_{2,w}. \tag{3.31}$$

That is, the optimal approximation $\tilde{V}^*(x)$ minimizes the weighted $\mathcal{L}_2$-norm of the approximation errors.

Let the approximation error, also known as the *residual*, take the form

$$R(x) = \mathcal{T}\left(\sum_{i=1}^{n} w_i \phi_i(x)\right) - l(x). \tag{3.32}$$

The Galerkin method requires that the error be orthogonal to a weight function in the form

$$W_i = \frac{\partial \tilde{V}(x)}{\partial w_i} = \phi_i(x), \tag{3.33}$$

which is simply the basis function, i.e.,

$$\int_a^b R_n(x)\phi_i(x)\,dx = 0, \quad \text{for} \quad i = 1, \ldots, n. \tag{3.34}$$

This produces

$$\int_a^b \phi_i(x)\mathcal{T}\left(\sum_{i=1}^{n} w_i \phi_i(x)\right) dx - \int_a^b \phi_i(x) l(x)\,dx = 0, \quad \text{for} \quad i = 1, \ldots, n. \tag{3.35}$$

In the case where the operator $\mathcal{T}(V)$ does not include time derivatives, the Galerkin method produces a set of n algebraic equations. This is the case in the Isaacs equation. Otherwise, it produces a set of n ordinary differential equations, with n unknown coefficients for the basis functions. In general, the integrals can be evaluated numerically with quadratures, and the resulting problem is then a nonlinear least-squares problem for the coefficients w_i.

We are now in a position to consider the computational issues in choosing an approximate representation for the value function. There are many ways to select a basis function. For simple problems with linear dynamics, directly using the dimensions of the states as basis functions would suffice, i.e.,

$$\begin{aligned} \phi_0(x) &= 1 \\ \phi_i(x) &= x_i. \end{aligned} \tag{3.36}$$

Clearly, this simple scheme cannot represent complex value functions for nonlinear systems. An immediate extension is using a k-th order polynomial as a basis in the form

$$\phi_i(x) = \prod_{j=1}^{n} x_j^{m_j}, \quad m_j = [1 \dots k], \tag{3.37}$$

but this choice is subject to Runge's phenomenon, as discussed in Section 2.1. For a broad overview of various function approximation schemes in the context of dynamic programming and reinforcement learning, we refer the reader to Kober [36].

In this work, we utilize Fourier approximations in connection with the spectral approximation framework presented in Chapter 2. As discussed in Section 2.5, Fourier basis functions enjoy the property of spectral accuracy, which results in the exponential convergence of the approximation when the order of the Fourier series is increased. As discussed in Konidaris [131], there are very few existing applications of Fourier approximation in dynamic programming and reinforcement learning.

We first consider Fourier approximation for single dimensional functions. Given a single-dimensional function $V(x)$, its n-th degree Fourier expansion with period T is a linear combination of sinusoidal functions.

Definition 3.5. (Single-dimensional Fourier expansion). For any single-dimensional

function $V(x)$, the Fourier expansion $\tilde{V}(x)$ takes the form

$$\tilde{V}(x) = \frac{a_0}{2} + \sum_{k=1}^{n} \left[a_k \cos\left(k\frac{2\pi}{T}x\right) + b_k \sin\left(k\frac{2\pi}{T}x\right) \right], \tag{3.38}$$

where the coefficients a_k and b_k are given by

$$\begin{aligned} a_k &= \frac{2}{T}\int_0^T V(x)\cos\left(k\frac{2\pi}{T}x\right)dx \\ b_k &= \frac{2}{T}\int_0^T V(x)\sin\left(k\frac{2\pi}{T}x\right)dx. \end{aligned} \tag{3.39}$$

For an unknown $V(x)$, we can treat the coefficients a_k and b_k as values to be learned in a linear function approximation using the basis ϕ_i in the form

$$\phi_i(x) = \begin{cases} 1 & i = 0 \\ \cos(\frac{i+1}{2}\pi x) & i > 0, \text{if } i \text{ is odd} \\ \sin(\frac{i}{2}\pi x) & i > 0, \text{if } i \text{ is even} \end{cases}. \tag{3.40}$$

For a multivariate function $V(x)$ with period T in m-dimensions, we have

Definition 3.6. (Multi-dimensional Fourier expansion). For any multi-dimensional function $V(x)$, the Fourier expansion $\tilde{V}(x)$ takes the form

$$\tilde{V}(x) = \sum_{c} \left[a_c \cos\left(k\frac{2\pi}{T}cx\right) + b_c \sin\left(k\frac{2\pi}{T}cx\right) \right] \tag{3.41}$$

with a vector of coefficients $c = [c_1, \ldots, c_m]$.

For an n-th order Fourier approximation in m-dimensions, this results in $2(n+1)^m$ basis functions, which is an exponential explosion of parameters in the number of dimensions. There are several ways that we can reduce the number of basis functions. First, we note

that $V(x)$ can be even if

$$V(x) = V(-x) \tag{3.42}$$

and odd if

$$V(x) = -V(-x). \tag{3.43}$$

In these cases, the symmetry in the function reduces the coefficients to $a_i = 0$ and $b_i = 0$, respectively. This means that the corresponding sin and cos terms can be dropped, which reduces the number of basis functions to $(n + 1)$ for a single dimension in the form

$$\phi_i(x) = \cos(\pi c \cdot x). \tag{3.44}$$

To handle the exponential nature of the growth, we use a common technique in function approximation called variable coupling, which removes the simultaneous contribution of multiple state variables. For example, in a two-dimensional problem with a polynomial basis function that contains the terms $1, x_1, x_2$, and x_1x_2, we assume that x_1 and x_2 contributes to the approximation independently and uncouple the variables by dropping the x_1x_2 term. In the context of Fourier basis functions, we obtain an uncoupled basis by requiring that only one element is c is non-zero. For a k-th order approximation with n states, this results in $n(k + 1)$ number of basis functions. Table 3.1 lists the coupled and uncoupled Fourier basis functions for a two-dimensional system with orders 1, 2, and 3.

3.4 Example - two-dimensional uncertain system

3.4.1 Dynamics

In this section, we solve an example $\mathcal{H}_\infty$ robust control problem under the differential game formulation, using the derivations of the necessary and sufficient conditions for optimality,

as well as parametric approximations to the value function.

Consider the robust control problem

$$J(x,u,w,\theta)=\min_{u\in\mathcal{U}}\max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}}\frac{1}{2}\int_{t_0}^{t_f}\left[\|z(t)\|^2-\gamma^2\|w(t)\|^2\right]dt, \tag{3.45}$$

subject to the dynamics

$$\begin{aligned}
\dot{x}_1(t)&=x_2(t)\\
\dot{x}_2(t)&=-\left(\frac{1}{2}+\theta(t)\right)x_2^5(t)-\frac{1}{2}x_2^3(t)-x_1(t)+x_2(t)w(t)+(1+\theta(t))\,u\\
z(t)&=[x_2(t),u(t)]^T\\
x(t_0)&=x_0\\
\gamma&=1,\quad \beta=1.
\end{aligned} \tag{3.46}$$

We can convert this system to the matrix control-affine form

$$\begin{aligned}
\dot{x}(t)&=[f(x)+\Delta f(x,\theta,t)]+g_1(x)w(t)+[g_2(x)+\Delta g_2(x,\theta,t)]\,u(t)\\
z(t)&=[h(x),u(t)]^T\\
x(t_0)&=x_0,
\end{aligned} \tag{3.47}$$

with $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$ in parameterized form

$$\begin{aligned}
\Delta f(x,\theta,t)&=H_2(x)F(x,\theta,t)E_1(x)\\
\Delta g_2(x,\theta,t)&=g_2(x)F(x,\theta,t)E_2(x),
\end{aligned} \tag{3.48}$$

and $||F(x,\theta,t)||^2 \leq 1$. The system matrices are given by

$$
\begin{aligned}
f(x) &= \begin{bmatrix} x_2(t) \\ -\frac{1}{2}x_2^5(t) - \frac{1}{2}x_2^3(t) - x_1(t) \end{bmatrix}, \quad g_1(x) = \begin{bmatrix} 0 \\ x_2(t) \end{bmatrix}, \quad g_2(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \\
H_2(x) &= \begin{bmatrix} 0 - x_2^2(t) \end{bmatrix} \\
E_1(x) &= x_2(t), \quad E_2(x) = 1, \quad h(x) = x_2(t), \quad F(x,\theta,t) = \theta(t)
\end{aligned} \tag{3.49}
$$

The uncertain term $\theta(t)$ can take on the form of any function (time-varying or not), assuming that the squared-norm of the function is bounded according to the assumptions given. An example is $\theta(t) = 0.01\sin(t)$, which can be interpreted as a small perturbation added to the nominal system parameter.

3.4.2 Necessary and sufficient conditions

Following the definitions in Section 3.2, the saddle-point equilibrium of the problem is given by the value function

$$
V(x,t) = \inf_{u\in\mathcal{U}} \sup_{\substack{w\in\mathcal{W} \\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(\tau)\|^2 - \gamma^2 \|w(\tau)\|^2 \right] d\tau, \tag{3.50}
$$

where

$$
V_{x_1}(x,t) = \frac{\partial V(x,t)}{\partial x_1}, \quad V_{x_2}(x,t) = \frac{\partial V(x,t)}{\partial x_2}, \quad V_t(x,t) = \frac{\partial V(x,t)}{\partial t}, \tag{3.51}
$$

are the partial derivatives of the value function with respect to the state variables. The necessary and sufficient condition for optimality is the Hamilton-Jacobi-Isaacs equation,

given by

$$V_x(x)f(x) + \frac{1}{2}V_x(x)\left[\frac{1}{\gamma^2}g_1(x)g_1^T(x) + H_2(x)H_2^T(x) \right.$$
$$\left. -g_2(x)\left(2I + \frac{1}{4}E_2^T(x)E_2(x)\right)g_2^T(x)\right]V_x^T(x) + \frac{1}{2}h^T(x)h(x) + \frac{1}{2}E_1(x)E_1^T(x) \leq 0. \tag{3.52}$$

Substituting the system matrices, we have

$$\frac{V_{x_2}^2(x)\ (x_2^4 + x_2^2 - 2)}{2} + V_{x_1}(x)\, x_2 + x_2^2 - V_{x_2}(x)\left(\frac{x_2^5}{2} + \frac{x_2^3}{2} + x_1\right) \leq 0. \tag{3.53}$$

By assuming a positive-definite value function candidate in the form

$$V(x) = \frac{1}{2}(x_1^2 + x_2^2) \tag{3.54}$$

with the costate

$$V_x(x) = [x_1 \quad x_2], \tag{3.55}$$

we can show that

$$\frac{x_2^2\ (x_2^4 + x_2^2 - 2)}{2} - x_2\left(\frac{x_2^5}{2} + \frac{x_2^3}{2} + x_1\right) + x_1\, x_2 + x_2^2 = 0, \tag{3.56}$$

which satisfies the HJI. Hence, the optimal controls are given by

$$u^*(t) = -\left(2I + \frac{1}{4}E_2^T(x)E_2(x)\right)^T g_2^T V_x^T(x)$$
$$w^*(t) = \frac{1}{\gamma^2}g_1^T(x)V_x^T(x), \tag{3.57}$$

which in this case is

$$u^*(t) = -2x_2$$
$$w^*(t) = x_2^2. \tag{3.58}$$

3.4.3 Value function approximation

We follow the approach presented in Section 3.3 to solve the problem (3.45), subject to the dynamics (3.46). For simple problems such as this, we can in fact solve the value function approximation problem almost analytically. In this example problem, we use both a traditional polynomial as well as the Fourier basis function.

Given the approximate value function $\tilde{V}(x)$ as a weighted sum of a set of basis functions ϕ_i

$$\tilde{V}(x) = \sum_{i=1}^{n} w_i \phi_i(x), \tag{3.59}$$

we first use the approximation

$$\tilde{V}(x) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 x_2^2 + w_4 x_1^2 + w_5 x_2^2 \tag{3.60}$$

with the derivative

$$\tilde{V}_x(x) = \begin{bmatrix} 2w_3 x_1 x_2^2 + w_1 + 2w_4 x_1, & 2w_3 x_2 x_1^2 + w_2 + 2w_5 x_2 \end{bmatrix}. \tag{3.61}$$

Let the residual function take the form

$$R(x) = \mathcal{T}\left(\sum_{i=1}^{n} w_i \phi_i(x) \right) - l(x). \tag{3.62}$$

In this case, this is simply the Isaacs equation evaluated with the approximated value

function (3.60), which results in

$$\begin{aligned} R(x) \; &= x_2 \, \left(2\, w_3\, x_1\, x_2^2 + w_1 + 2\, w_4\, x_1\right) + x_2^2 \\ &\quad - \left(\frac{x_2^5}{2} + \frac{x_2^3}{2} + x_1\right) \, \left(2\, w_3\, x_2\, x_1^2 + w_2 + 2\, w_5\, x_2\right) \\ &\quad + \left(x_2^4 + x_2^2 - 2\right) \, \left(2\, w_3\, x_2\, x_1^2 + w_2 + 2\, w_5\, x_2\right) \, \left(w_3\, x_2\, x_1^2 + \frac{w_2}{2} + w_5\, x_2\right). \end{aligned} \tag{3.63}$$

Immediately, we notice that w_0 disappears. By setting

$$\tilde{V}(0) = 0, \tag{3.64}$$

we also find that $-w_2^2 = 0$, hence $w_2 = 0$. To find the remaining weights w_1, w_3, w_4, w_5, we generate the system of equations

$$\int_a^b R_n(x)\phi_i(x)\, dx = 0, \quad \text{for} \quad i = 1, \ldots, n. \tag{3.65}$$

This produces four equations

$$\begin{aligned} \int_0^1 \int_0^1 R(x) \Big(w_1 x_1\Big)\, dx_1\, dx_2 &= 0 \\ \int_0^1 \int_0^1 R(x) \Big(w_3 x_1^2 x_2^2\Big)\, dx_1\, dx_2 &= 0 \\ \int_0^1 \int_0^1 R(x) \Big(w_4 x_1^2\Big)\, dx_1\, dx_2 &= 0 \\ \int_0^1 \int_0^1 R(x) \Big(w_5 x_2^2\Big)\, dx_1\, dx_2 &= 0. \end{aligned} \tag{3.66}$$

Normally, these integrals would be evaluated numerically with quadratures, but in this

case, the equations are simple enough to solve analytically, which results in

$$\begin{pmatrix} -\frac{34\,{w_3}^2}{315} - \frac{34\,w_3\,w_5}{105} - \frac{5\,w_3}{42} - \frac{34\,{w_5}^2}{105} - \frac{53\,w_5}{105} + \frac{w_1}{4} + \frac{w_4}{3} + \frac{1}{6} \\ -\frac{92\,{w_3}^2}{2205} - \frac{184\,w_3\,w_5}{1575} - \frac{16\,w_3}{315} - \frac{92\,{w_5}^2}{945} - \frac{317\,w_5}{1512} + \frac{w_1}{12} + \frac{w_4}{8} + \frac{1}{15} \\ -\frac{68\,{w_3}^2}{735} - \frac{136\,w_3\,w_5}{525} - \frac{463\,w_3}{4200} - \frac{68\,{w_5}^2}{315} - \frac{51\,w_5}{140} + \frac{w_1}{6} + \frac{w_4}{4} + \frac{1}{9} \\ -\frac{92\,{w_3}^2}{1575} - \frac{184\,w_3\,w_5}{945} - \frac{65\,w_3}{1512} - \frac{92\,{w_5}^2}{315} - \frac{127\,w_5}{252} + \frac{w_1}{4} + \frac{w_4}{4} + \frac{1}{5} \end{pmatrix} = 0, \tag{3.67}$$

which yields the solution $w_1 = 0$, $w_3 = 0$, $w_4 = 0.5$, and $w_5 = 0.5$, i.e.,

$$V(x) = \frac{1}{2}(x_1^2 + x_2^2). \tag{3.68}$$

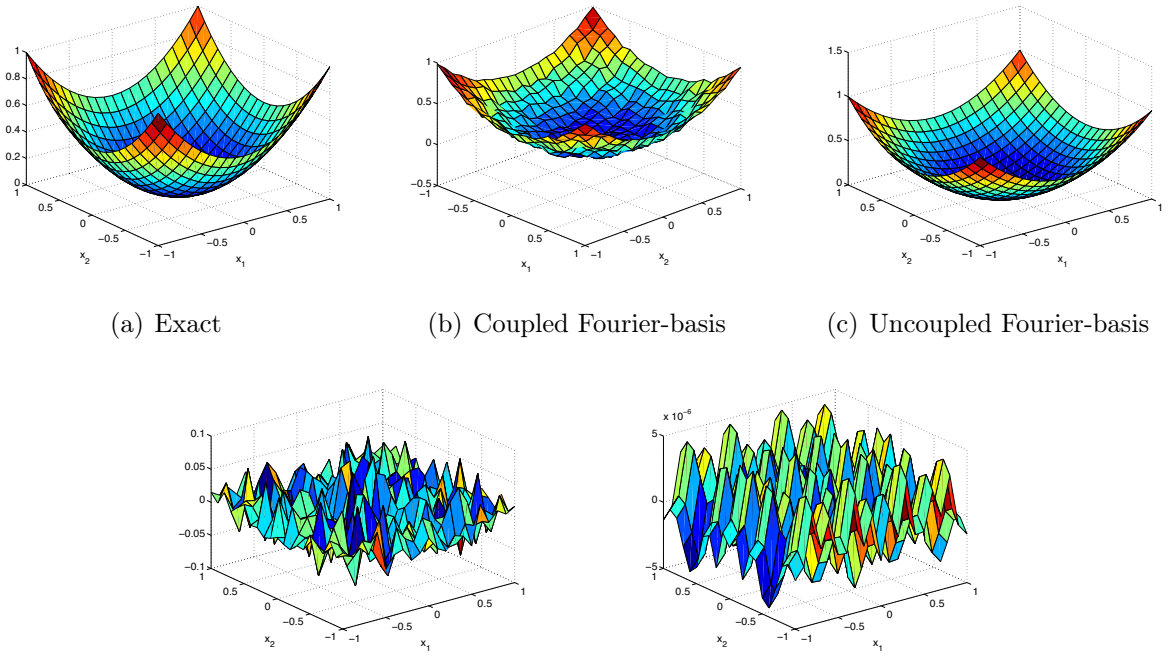
Now let us consider a more general case of value function approximation in the form

$$\tilde{V}(x) = \sum_{i=1}^{n} w_i \phi_i(x), \tag{3.69}$$

with $\phi_i(x)$ given by Fourier basis functions. In this case, the system of integral equations must be solved numerically, with a Gaussian quadrature embedded within a trust region dogleg iteration to solve the system of equations.

To approximate the value function, we used the Fourier-basis functions described in Section 3.3 in both coupled and uncoupled settings. For a 15-degree Fourier basis approximation of a two-dimensional system, there are 256 coupled basis functions and 31 uncoupled basis functions. Because the exact form of the value function is in fact uncoupled, we expect that the uncoupled Fourier basis will converge faster.

We searched the coefficients for the basis functions for 500 iterations and generated Fig. 3.1. Indeed, the coupled Fourier basis Fig. 3.1(b) was slower to produce the same approximation compared to the uncoupled basis Fig. 3.1(c). Using the same number of iterations, the approximation quality for the coupled basis is four orders of magnitude worse than the uncoupled. When compared with the exact value function Fig. 3.1(a),



(a) Exact

(b) Coupled Fourier-basis

(c) Uncoupled Fourier-basis

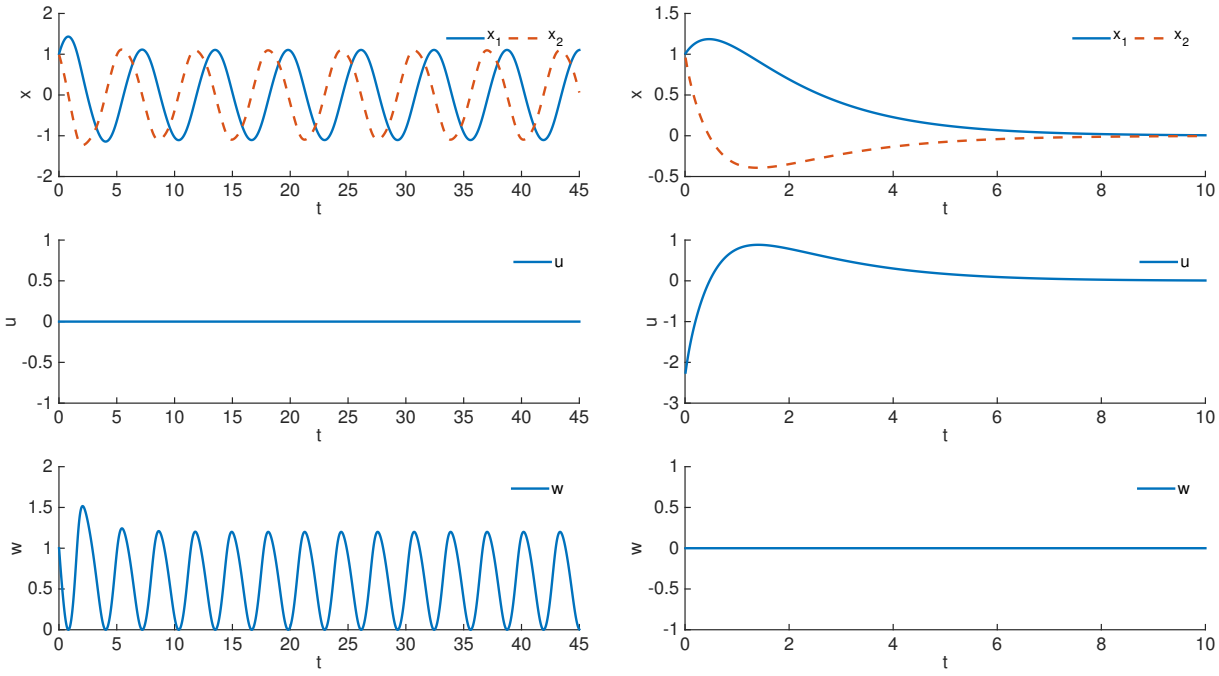
(d) Coupled Fourier basis residuals

(e) Uncoupled Fourier basis residuals. Note the changed vertical scale compared to Fig. 3.1(d).

Figure 3.1: Exact value function, Fourier-basis approximation, and their residuals.

the residuals of the coupled and uncoupled approximations are given in Fig. 3.1(d) and Fig. 3.1(e).

Using the initial condition $x_0 = [1 \quad 1]^T$, we can simulate the open-loop (without controls) response of the system under external disturbance and verify that the system is unstable, as shown in Fig. 3.2(a). Using the control laws (3.19) in conjunction with the Fourier approximated value function in Fig. 3.1(c), the closed-loop response of the system is shown in Fig. 3.2(b), assuming no external disturbance and no parametric uncertainties. This is consistent with the theoretical analysis under Lyapunov stability, where we showed in (**??**) that using a value function as a Lyapunov candidate, the resulting system is stable



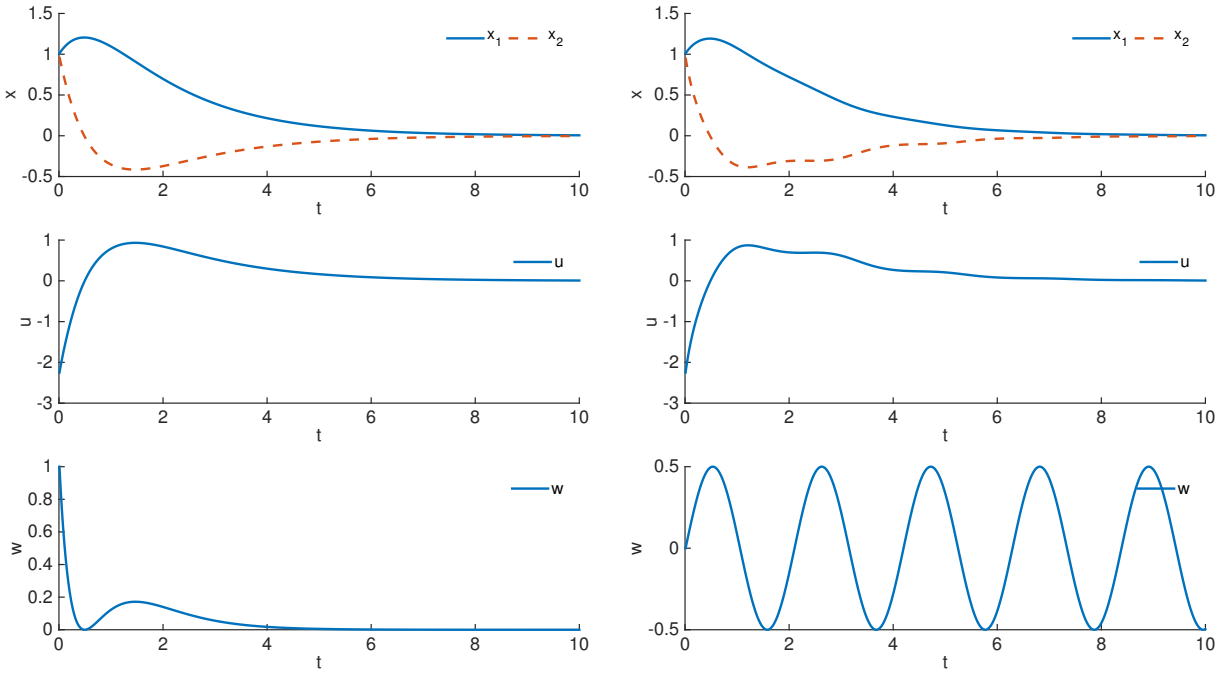
(a) Open-Loop response.

(b) Closed-Loop response with no disturbance.

Figure 3.2: Open and closed-loop response of a two-dimensional system under disturbance. Without controls, the system does not converge. With controls but without disturbances and uncertainties, the system is stable.

with and without disturbances.

We can further verify the performance of the control design under bounded external disturbances and parametric uncertainties. With $\theta(t) = 0.01\sin(t)$, Fig. 3.3 demonstrates the closed-loop response of the system. Fig. 3.3(a) shows the response of the system under optimal disturbance, with $w(t)$ computed from (3.19), and Fig. 3.3(b) shows the response under a non-optimal disturbance in the form of a sinusoidal signal. In both cases, the feedback control law was able to stabilize the systems given the disturbances and uncertainties.



(a) Closed-loop response with optimal disturbance.

(b) Closed-loop response with sinusoidal disturbance.

Figure 3.3: Closed-loop response of a two-dimensional system. The feedback control is able to stabilize the system under both optimal and sinusoidal disturbances.

3.5 Summary

In this chapter, we've derived necessary and sufficient conditions for optimality for a $\mathcal{H}_\infty$ robust control problem in terms of a Hamilton-Jacobi-Isaacs equation. A value function was given as the solution of the Isaacs equation, from which optimal feedback control laws were derived for both players. For practical solutions of the Isaacs equation, we presented a value function approximation approach, which relied on Fourier basis functions in connection with Galerkin's method, under the general spectral approximation framework presented in Chapter 2.5.

Ultimately, for systems of complex nonlinear behaviors and increased dimensionality, the scalability of value function approximation is limited. One effective way of avoiding the exponential explosion of computational demand associated with searching for global value functions is to only search for *locally optimal* solutions. In the next chapter, we explore the class of trajectory optimization techniques to solve larger-scale problems which out of the reach of function approximation-based methods.

Table 3.1: Coupled and uncoupled Fourier basis functions for a two-dimensional system.

Approximation order n	Coupled	Uncoupled
1	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(\pi\, x_1) \\ \cos(\pi\, x_1 + \pi\, x_2) \end{pmatrix}$	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(\pi\, x_1) \end{pmatrix}$
2	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(2\,\pi\, x_2) \\ \cos(\pi\, x_1) \\ \cos(\pi\, x_1 + \pi\, x_2) \\ \cos(\pi\, x_1 + 2\,\pi\, x_2) \\ \cos(2\,\pi\, x_1) \\ \cos(2\,\pi\, x_1 + \pi\, x_2) \\ \cos(2\,\pi\, x_1 + 2\,\pi\, x_2) \end{pmatrix}$	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(2\,\pi\, x_2) \\ \cos(\pi\, x_1) \\ \cos(2\,\pi\, x_1) \end{pmatrix}$
3	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(2\,\pi\, x_2) \\ \cos(3\,\pi\, x_2) \\ \cos(\pi\, x_1) \\ \cos(\pi\, x_1 + \pi\, x_2) \\ \cos(\pi\, x_1 + 2\,\pi\, x_2) \\ \cos(\pi\, x_1 + 3\,\pi\, x_2) \\ \cos(2\,\pi\, x_1) \\ \cos(2\,\pi\, x_1 + \pi\, x_2) \\ \cos(2\,\pi\, x_1 + 2\,\pi\, x_2) \\ \cos(2\,\pi\, x_1 + 3\,\pi\, x_2) \\ \cos(3\,\pi\, x_1) \\ \cos(3\,\pi\, x_1 + \pi\, x_2) \\ \cos(3\,\pi\, x_1 + 2\,\pi\, x_2) \\ \cos(3\,\pi\, x_1 + 3\,\pi\, x_2) \end{pmatrix}$	$\phi(x) = \begin{pmatrix} 1 \\ \cos(\pi\, x_2) \\ \cos(2\,\pi\, x_2) \\ \cos(3\,\pi\, x_2) \\ \cos(\pi\, x_1) \\ \cos(2\,\pi\, x_1) \\ \cos(3\,\pi\, x_1) \end{pmatrix}$

4

Robust Trajectory Optimization

4.1 Motivation

In the previous chapter, we have examined $\mathcal{H}_\infty$ robust control from the perspective of dynamic programming and derived solutions based on necessary and sufficient conditions for optimality given parametric uncertainty and external disturbances. We derived global solutions to this problem using the exact Bellman value function as well as its parametric approximations. Though the approach characterized global solutions, its practical implementation is limited by the complexity of the underlying dynamic system.

It is well-known in optimal control that bypassing the curse of dimensionality is a difficult endeavor. In the value function approximation framework in Section 3.3, we formulated the notion of decoupled basis functions, which mitigates the effects of the curse of dimensionality. Ultimately, for systems of complex nonlinear behaviors and increased dimensionality, the scalability of value function approximation is limited. One effective way of avoiding the exponential explosion of computational demand associated with searching for global value functions is to only search for *locally optimal* solutions.

Trajectory optimization refers to methods where we seek to obtain only open-loop trajectories that satisfy certain optimality conditions. Approaches to trajectory optimization can be divided into *direct* and *indirect* methods, based on the order in which necessary conditions for optimality are applied. These methods are reviewed in Section 1.2.5. By

bypassing the need to solve the full necessary and sufficient conditions for optimality, trajectory optimization is a viable approach for avoiding the curse of dimensionality. However, while direct trajectory optimization has been successful in addressing optimal control problems, applying it to robust control is not straightforward and a naive implementation tends to result in an expensive optimization problem (discussed in Section 1.2.5).

In this chapter, we extend direct trajectory optimization to $\mathcal{H}_\infty$ control and differential games by transforming it into a mixed complementarity problem (MCP). This approach is consistent with the spectral approximation methods presented in Chapter 2, and the resulting discrete optimization problem can be solved efficiently using commercially available solvers. Consequently, the resulting approach is suitable to be applied in a receding-horizon control setting, which utilizes open-loop trajectories in a closed-loop setting. We show that the approach is robust against bounded external disturbances and parametric uncertainties, and demonstrate the practical implementation of the approach using a simple two-dimensional system, then compare the results against the global solutions obtained in the previous chapter.

4.2 First-order necessary conditions

We begin the formulation of our trajectory optimization method by first discussing the indirect approach. In Section 1.3, we formulated a robust $\mathcal{H}_\infty$-control problem with the minimax objective

$$J(x,u,w,\theta) = \min_{u\in\mathcal{U}} \max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2\|w(t)\|^2\right] dt, \tag{4.1}$$

which entails a two-player, noncooperative, zero-sum differential game over a horizon $t_f > t_0$, subject to the nonlinear dynamics

$$\begin{aligned} \dot{x}(t) &= \Big[f(x) + \Delta f(x,\theta,t)\Big] + g_1(x)w(t) + \Big[g_2(x) + \Delta g_2(x,\theta,t)\Big]u(t) \\ z(t) &= [h(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{4.2}$$

with $x(t) \in \mathcal{X}$ as the *state* vector, $z(t) \in \mathcal{Z}$ is the *output* vector, $u(t) \in \mathcal{U}$ and $w(t) \in \mathcal{W}$ are the minimizing and maximizing players, respectively, x_0 is a vector of *initial states*, and the matrices $f(x)$, $g(x)$, and $h(x)$ are the *system matrices* that are smooth vector fields not explicitly parameterized by time. The parameter $\theta \in \Theta$ is the uncertain system parameter which belongs to the set

$$\Theta = \{\theta | 0 \leq \theta \leq \theta_u\} \tag{4.3}$$

that is bounded above by θ_u. Parametrized by θ, the functions $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$ are uncertain functions belonging to the set of *admissible uncertainties*.

In the settings of the two-player differential game, open-loop solutions entail that both players formulate their strategy at the moment the system starts to evolve, based on information available at the time: the system dynamics, the objective function, and initial conditions. This strategy cannot be changed once the system evolves and the saddle-point solution is the combination of strategies of both players which are secured against any attempt by one player to unilaterally change his strategy.

Let us first consider a slightly more general version of this problem, with a more general form of the objective function and system dynamics as compared to (4.1) and (4.2).

Definition 4.1. (Objective function - primal). The optimization objective takes the form

$$J(x,u,w,\theta,t_0,t_f) = \Phi(x(t_0),x(t_f),t_0,t_f) + \int_{t_0}^{t_f} C(x,u,w)\,dt, \tag{4.4}$$

where $\Phi(x(t_0), x(t_f), t_0, t_f)$ is commonly referred to as the *terminal cost* and $C(x, u, w)$ is the *one-step cost* integrated over the horizon $t_f > t_0$. We assume that $x(t) = 0$ is an unique equilibrium point of the system with $u(t) = 0$ and $w(t) = 0$.

Definition 4.2. (Uncertain system dynamics). The dynamics of the system is given by a set of first-order differential equations

$$\dot{x}(t) = f(x, u, w, \theta), \tag{4.5}$$

where x is the *state* vector, u is the vector of controlled inputs to the system, w is the vector of external disturbances, and *theta* is the time-varying unknown parameters within the system. For this set of dynamics, the initial condition and final condition are expressed as

$$e(x(t_0), x(t_f), t_0, t_f) = 0, \tag{4.6}$$

which is a boundary constraint at both the initial time t_0 and final time t_f. Additional path constraints take the form of an inequality

$$s(x, u, w) \leq 0, \tag{4.7}$$

which represent constraints on the states and control inputs. We assume that the constraints on the states and control inputs are fixed and contains no uncertainties. We assume that f, Φ, C, e, and s are nonlinear and smooth functions with respect to x, u, and w.

The optimization (4.4), subject to the dynamics (4.5), boundary conditions (4.6), and path constraints (4.7) is called the *primal problem*. We now convert this constrained optimization problem into an unconstrained problem in the *dual* form. Let us define a Lagrange multiplier v to combine the terminal cost (4.4) and terminal constraint (4.6) in

the form

$$E(x(t_0), x(t_f), t_0, t_f, v) = \Phi(x(t_0), x(t_f), t_0, t_f) + v^T e(x(t_0), x(t_f), t_0, t_f) \tag{4.8}$$

and define the Lagrange multipliers λ and μ to incorporate the dynamics (4.5) and path constraints (4.7) to form the objective function

$$\begin{aligned} J(x, u, w, \theta, t_0, t_f) &= E(x(t_0), x(t_f), t_0, t_f, v) \\ &+ \int_{t_0}^{t_f} \Big[C(x, u, w) + \lambda^T \Big(f(x, u, w, \theta) - \dot{x}(t) \Big) + \mu^T s(x, u, w) \Big] \, dt. \end{aligned} \tag{4.9}$$

Similar to (3.8), we use the *Hamiltonian* to express the concatenation of the various optimization constraints.

Definition 4.3. (Hamiltonian). The Hamiltonian $H(x, u, w, \theta, \lambda, \mu)$ for the system dynamics in (4.5) and objective function (4.9) is a function of the states x, control inputs u and w, as well as the Lagrange multipliers λ and μ in the form

$$H(x, u, w, \theta, \lambda, \mu) = C(x, u, w) + \lambda^T \Big(f(x, u, w, \theta) \Big) + \mu^T \Big(s(x, u, w) \Big), \tag{4.10}$$

where $C(x, u, w)$ is the one step cost, $f(x, u, w)$ is the system dynamics , and $s(x, u, w)$ is the path constraint set.

In this context, Lagrange multipliers v, λ, and μ are commonly referred to as the *costate* or *adjoint* variables. Substituting the Hamiltonian into the objective function, we have

$$J(x, u, w, \theta, t_0, t_f) = E(x(t_0), x(t_f), t_0, t_f, v) + \int_{t_0}^{t_f} \big[H(x, u, w, \theta, \lambda, \mu) - \lambda^T \dot{x}(t) \big] \, dt. \tag{4.11}$$

Examining the last term in the integral, we can express λ in terms of its derivative $\dot{\lambda}$ as

$$\begin{aligned}\int_{t_0}^{t_f} \lambda^T(t)\dot{x}(t)\,dt &= \lambda^T(t)x(t)|_{t_0}^{t_f} - \int_{t_0}^{t_f} \dot{\lambda}^T(t)x(t)\,dt \\ &= \lambda^T(t_f)x(t_f) - \lambda^T(t_0)x(t_0) - \int_{t_0}^{t_f} \dot{\lambda}^T(t)x(t)\,dt. \end{aligned} \tag{4.12}$$

Effectively, this gives us the dynamics of λ with respect to the state x, at the initial and final times t_0 and t_f, respectively. Rewriting the objective function, we obtain

Definition 4.4. (Objective function - dual) The optimization objective for the dual problem takes the form

$$\begin{aligned}J(x,u,w,\theta,t_0,t_f) =& E(x(t_0),x(t_f),t_0,t_f,v) - \lambda^T(t_f)x(t_f) \\ &+ \lambda^T(t_0)x(t_0) + \int_{t_0}^{t_f} \left[H(x,u,w,\theta,\lambda,\mu) + \dot{\lambda}^T(t)x(t)\right]\,dt, \end{aligned} \tag{4.13}$$

where $E(x(t_0),x(t_f),t_0,t_f,v)$ is the terminal cost, λ is the costate, and $H(x,u,w,\theta,\lambda,\mu)$ is the Hamiltonian.

Given this augmented objective function with constraints adjoined in the manner of Lagrange, we are now in a position to derive the first-order necessary conditions for optimality using the calculus of variations. The first variation of J is

$$\begin{aligned}\delta J(x,u,w,\theta,t_0,t_f) =& \frac{\partial E}{\partial x(t_f)}dx(t_f) + \frac{\partial E}{\partial t_f}\delta t_f - \lambda^T(t_f)\delta x(t_f) + \int_{t_f}^{t_f+\delta t_f} C(x,u,w)\,dt \\ &+ \int_{t_0}^{t_f} H_x(x,u,w,\lambda,\mu)\delta x\,dt + \int_{t_0}^{t_f} H_u(x,u,w,\lambda,\mu)\delta u\,dt \\ &+ \int_{t_0}^{t_f} H_w(x,u,w,\lambda,\mu)\delta w\,dt + \int_{t_0}^{t_f} H_\theta(x,u,w,\lambda,\mu)\delta\theta\,dt + \int_{t_0}^{t_f} \dot{\lambda}^T\delta x\,dt = 0, \end{aligned} \tag{4.14}$$

where the various derivatives are defined as

$$\begin{aligned}
\frac{\partial E}{\partial x(t_f)} &= \left[\frac{\partial E}{\partial x_1(t_f)}, \ldots, \frac{\partial E}{\partial x_{N_x}(t_f)}\right] \\
dx(t_f) &= \delta x(t_f) + \dot{x}(t_f)\delta t_f = \delta x(t_f) + f(x,u,w,\theta)\Big|_{t=t_f}\delta t_f \\
H_x(x,u,w,\theta,\lambda,\mu) &= \left[\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial x_1}, \ldots, \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial x_{N_x}}\right]^T \\
H_u(x,u,w,\theta,\lambda,\mu) &= \left[\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial u_1}, \ldots, \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial u_{N_u}}\right]^T \\
H_w(x,u,w,\theta,\lambda,\mu) &= \left[\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial w_1}, \ldots, \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial w_{N_u}}\right]^T \\
H_\theta(x,u,w,\theta,\lambda,\mu) &= \left[\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial \theta_1}, \ldots, \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial \theta_{N_\theta}}\right]^T.
\end{aligned} \tag{4.15}$$

Since

$$\int_{t_f}^{t_f+\delta t_f} C(x,u,w)\,dt = C|_{t=t_f}\delta t_f, \tag{4.16}$$

we can simplifying and collect the terms,

$$\begin{aligned}
\delta J = &\left[\frac{\partial E}{\partial x(t_f)} - \lambda^T(t_f)\right]\delta x(t_f) + \left[\frac{\partial E}{\partial t_f} + \frac{\partial E}{\partial x(t_f)} f(x,u,w,\theta)\Big|_{t=t_f} + C(x,u,w)\Big|_{t=t_f}\right]\delta t_f \\
&+ \int_{t_0}^{t_f} (H_x(x,u,w,\theta,\lambda,\mu) + \dot{\lambda}^T)\delta x\,dt + \int_{t_0}^{t_f} H_u(x,u,w,\theta,\lambda,\mu)\delta u\,dt \\
&+ \int_{t_0}^{t_f} H_w(x,u,w,\theta,\lambda,\mu)\delta w\,dt + \int_{t_0}^{t_f} H_\theta(x,u,w,\theta,\lambda,\mu)\delta\theta\,dt = 0.
\end{aligned} \tag{4.17}$$

The extremum of the function J is obtained when the first variation of δJ vanishes, i.e.,

$$\delta J(x,u,w,\theta,t_0,t_f) = 0 \tag{4.18}$$

and hence the necessary conditions for optimality can be established by setting the coefficients of the independent variables δt_f, δx, δu, and δw to zero. This leads to

Theorem 4.5. (First-order necessary conditions for optimality). For the optimization problem (4.4), subject to the system dynamics (4.5), boundary conditions (4.6), and path constraints (4.7), the first-order necessary conditions for optimality is given by the stationary conditions for the Hamiltonian

$$\begin{aligned}\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial u} &= 0\\ \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial w} &= 0\\ \frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial \theta} &= 0,\end{aligned} \tag{4.19}$$

the costate equations

$$\begin{aligned}\dot{\lambda}^T &= -\frac{\partial H(x,u,w,\theta,\lambda,\mu)}{\partial x}\\ \lambda^T(t_f) &= \frac{\partial E}{\partial x(t_f)},\end{aligned} \tag{4.20}$$

and

$$\frac{\partial E}{\partial t_f} + \frac{\partial E}{\partial x(t_f)} f(x,u,w,\theta)\Big|_{t=t_f} + C(x,u,w)\Big|_{t=t_f} = 0. \tag{4.21}$$

We omit the proof here as it is trivial to verify that the choice of the costate enforces $\delta J(x,u,w,\theta,t_0,t_f) = 0$.

Intuitively, these results are consistent with the costate equations in Pontryagin's minimum principle for traditional optimal control problems. The optimal controls u^*, w^*, and θ^* are found by minimizing and maximizing the Hamiltonian, that is,

$$\begin{aligned}u^*(t) &= \arg\min_u H(x,u,w,\theta,\lambda,\mu)\\ w^*(t) &= \arg\max_w H(x,u,w,\theta,\lambda,\mu)\\ \theta^*(t) &= \arg\max_\theta H(x,u,w,\theta,\lambda,\mu).\end{aligned} \tag{4.22}$$

These control laws are similar to (3.11) with an important distinction that we had previously used $\lambda = V_x(x)$, where $V_x(x)$ is the gradient of the value function. Here, as value of the costate is obtained by solving the costate equation (4.20), the resulting two-point boundary value problem yields open-loop solutions to the optimization problem (4.4).

Beyond systems with very simple dynamics, the two-point boundary value problem in (4.20) must be solved numerically, via the so-called shooting methods with an initial guess for the ODE. This is a major drawback in applying the indirect method - the quality of the solution is strongly dependent on the quality of the initial guess.

4.3 Direct trajectory optimization

To improve computational efficiency in large-scale problems, we wish to avoid solving the necessary and sufficient condition for optimality in terms of the Isaacs equation and instead solve this problem using a direct numerical optimization approach. However, the Isaacs equation addressed an important challenge in the formulation of the solution: it combined the inner maximization problem in (4.1) in the Hamiltonian condition (3.11), and the resulting inequality removed the need to address the maximization problem directly. If we bypass the Isaacs equation, then we must find alternative ways to address the maximization operator.

To accomplish this, we rely on an algebraic manipulation which transforms the minimax optimization problem into a minimization problem with an inequality constraint bounded by the maximum value of the original maximization problem. Given the objective function

$$J(x,u,w,\theta) = \min_{u\in\mathcal{U}} \max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2\|w(t)\|^2\right] dt, \tag{4.23}$$

let m^* bound the maximum value attained by the inner maximization problem, i.e.,

$$m^* = \max_{\substack{w \in \mathcal{W} \\ \theta \in \Theta}} \frac{1}{2} \int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2 \|w(t)\|^2 \right] dt. \tag{4.24}$$

Effectively, this can be viewed as an inequality constraint

$$m^* \geq \frac{1}{2} \int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2 \|w(t)\|^2 \right] dt. \tag{4.25}$$

As a direct consequence, we can rewrite the original minimax optimization (4.1) as a minimization problem

$$J(x, u, w, \theta) = \min_{u \in \mathcal{U}} m^*, \tag{4.26}$$

subject to the constraint (4.25), which ensures that the maximum value attained by the disturbance player w is captured in m^*, which is then minimized to yield the solution to the minimizing player u. In this form, (4.25) is referred to as an isoperimetric constraint, since it's an inequality constraint with an integral term. Isoperimetric constraints are common in the optimal control literature, most famously in the Problem of Queen Dido [14]. An isoperimetric can be treated as any other inequality constraint and we will derive the necessary conditions for optimality in the subsequent sections.

Similar to direct trajectory optimization for optimal control problems, we aim to directly discretize the trajectories in our robust control problem. This is done in connection with the least-squares function approximation framework discussed in Chapter 2. We approximate the state, input, and disturbance trajectories with polynomials in a least-squares

sense, i.e., for every trajectory x_i, u_i, and w_i, there exists a polynomial that satisfies

$$\begin{aligned}
\|x_i(t) - \tilde{x}_i^*(t)\|_{2,q} &= \min_{\tilde{x}_i(t)} \|x_i(t) - \tilde{x}_i(t)\|_{2,q} \\
\|u_i(t) - \tilde{u}_i^*(t)\|_{2,q} &= \min_{\tilde{u}_i(t)} \|u_i(t) - \tilde{u}_i(t)\|_{2,q} \\
\|w_i(t) - \tilde{w}_i^*(t)\|_{2,q} &= \min_{\tilde{w}_i(t)} \|w_i(t) - \tilde{w}_i(t)\|_{2,q},
\end{aligned} \tag{4.27}$$

defined on the weighted $\mathcal{L}_2$-norm of a function $x_i(t)$

$$\|x_i(t)\|_{2,q} = \left(\int_a^b (x_i(t))^2 q(t)\, dt \right)^{\frac{1}{2}}, \tag{4.28}$$

over some weight $q(x)$. The trajectories $\tilde{u}_i^*(\tau)$ and $\tilde{w}_i^*(\tau)$ are defined similarly and we only describe the state trajectory approximation below for brevity.

To use this framework in approximate trajectories, we first discretize the time index from $[t_0, t_f]$ to $[-1, 1]$. As shown in (2.36), this can be accomplished through a simple linear transformation

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f - t_0}{2}. \tag{4.29}$$

Let $(\tau_1, \ldots, \tau_n)$ be the n collocation points in the domain $[-1, 1]$. The state trajectory $\tilde{x}_i^*(\tau)$ is a linear combination of basis functions in the form

$$\tilde{x}_i^*(\tau) = \sum_{i=1}^{N} c_i \phi_i(\tau), \tag{4.30}$$

with basis functions $\phi_i(\tau)$ and coefficients c_i. We use Legendre polynomials [100] as the basis functions, orthogonal with respect to $q(\tau)$ defined on the inner product

$$\int_a^b \phi_i(\tau)\phi_j(\tau)q(\tau)\, dx = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}, \tag{4.31}$$

where δ_{ij} is the Kronecker delta function. The corresponding coefficients are given by

$$c_i^* = \int_a^b \phi_i(\tau)x(\tau)q(\tau)\,d\tau, \quad i = 0, \ldots, n, \tag{4.32}$$

which can be solved using Gaussian quadrature, resulting in the Legendre-Gauss-Lobatto (LGL) collocation points defined on the interval $\tau \in [-1, 1]$ [98]. Given a set of trajectories $x_i(\tau)$, we use the notation C_N to denote a matrix of coefficients, where each row contains coefficients c_i for a single trajectory under N collocation points. The objective of the discrete optimization is to search this collection of coefficients to generate trajectories that satisfy the state dynamics and associated constraints, i.e.,

$$\min \psi(C_N), \tag{4.33}$$

subject to

$$\Theta(C_N) \leq 0, \quad \Gamma(C_N) = 0, \quad C_N \in B, \tag{4.34}$$

where $\Theta(C_N)$ and $\Gamma(C_N)$ represents concatenated inequality and equality constraints, respectively, and $B = \{C_N \in \mathbb{R}^n | r \leq C_N \leq s\}$ with $r_i \in [-\infty, \infty]$ and $s_i \in [r_i, \infty]$. Let $S = \{C_N \in B | \Theta(C_N) \leq 0, \Gamma(C_N) = 0\}$ denote the feasible region. The Lagrangian is defined as

$$L(C_N, \lambda, v) = \psi(C_N) - \lambda^T \Theta(C_N) - v^T \Gamma(C_N), \tag{4.35}$$

where λ and v denote the Lagrange multipliers for the constraints. According to the Karush-Kuhn-Tucker (KKT) conditions [132], the first order necessary conditions are

$$\begin{aligned} &0 \in \lambda_{C_N} L(C_N, \lambda, v) + N_b(C_N) \\ &0 \geq \lambda \perp \Theta(C_N) \leq 0 \\ &\Gamma(C_N) = 0, \end{aligned} \tag{4.36}$$

where $N_b(C_N) = \{z \in \mathbb{R}^n | (y - C_N)^T z \leq 0, \forall y \in B$ is the normal cone to B at C_N. In the case that r_i or s_i is finite, we can rewrite as

$$\begin{cases} (\nabla_{C_N} L(C_N, \lambda, v))_i \geq 0 & x_i = r_i \\ (\nabla_{C_N} L(C_N, \lambda, v))_i \leq 0 & x_i = s_i \\ (\nabla_{C_N} L(C_N, \lambda, v))_i = 0 & r_i < x_i < s_i. \end{cases} \tag{4.37}$$

A mixed complementarity problem [133] is defined as the problem of finding a point $z \in \mathbb{R}^n$ inside the box $B = \{z | -\infty \leq l < z < u \leq \infty\}$ that is complementary to a nonlinear function $F(z)$ when

$$\begin{cases} F_i(z) \geq 0 & z_i = l_i \\ F_i(z) \leq 0 & z_i = u_i \\ F_i(z) = 0 & l_i < z_i < u_i. \end{cases} \tag{4.38}$$

When $l = -\infty$ and $u = \infty$, the solution is given by simply solving a system of nonlinear equations so that $F(z) = 0$. If $l = 0$ and $u = \infty$, the resulting problem is a Nonlinear Complementarity Problem (NCP) such that

$$\begin{aligned} z_i &\geq 0 \\ F_i(z) &\geq 0 \\ z_i F_i(z) &= 0. \end{aligned} \tag{4.39}$$

The nonlinear MCP function can be written as a vector

$$F(z) = \begin{bmatrix} \nabla_{C_N} L(z) \\ \Theta(C_N) \\ \Gamma(C_N) \end{bmatrix}, l = \begin{bmatrix} r \\ -\infty \\ -\infty \end{bmatrix}, u = \begin{bmatrix} s \\ 0 \\ \infty \end{bmatrix} \tag{4.40}$$

where

$$\nabla_{C_N} L(z) = \nabla_{C_N} \psi(C_N) - \lambda^T \nabla_{C_N} \Theta(C_N) - v^T \nabla_{C_N} \Gamma(C_N). \tag{4.41}$$

This is an equivalent condition to the first order KKT conditions of the NLP, which can be solved by a MCP solver such as KNITRO [134].

4.4 Example - two-dimensional uncertain system

In this section, we solve an example $\mathcal{H}_\infty$ robust control problem under the differential game formulation using trajectory optimization. Previously in Section 3.4, we considered the robust control problem

$$J(x, u, w, \theta) = \min_{u \in \mathcal{U}} \max_{\substack{w \in \mathcal{W} \\ \theta \in \Theta}} \frac{1}{2} \int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2 \|w(t)\|^2 \right] dt, \tag{4.42}$$

subject to the dynamics

$$\begin{aligned} \dot{x}(t) &= [f(x) + \Delta f(x, \theta, t)] + g_1(x) w(t) + [g_2(x) + \Delta g_2(x, \theta, t)]\, u(t) \\ z(t) &= [h(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{4.43}$$

with $\Delta f(x, \theta, t)$ and $\Delta g_2(x, \theta, t)$ in parameterized form

$$\begin{aligned} \Delta f(x, \theta, t) &= H_2(x) F(x, \theta, t) E_1(x) \\ \Delta g_2(x, \theta, t) &= g_2(x) F(x, \theta, t) E_2(x), \end{aligned} \tag{4.44}$$

and $||F(x,\theta,t)||^2 \leq \beta$. The system matrices are given by

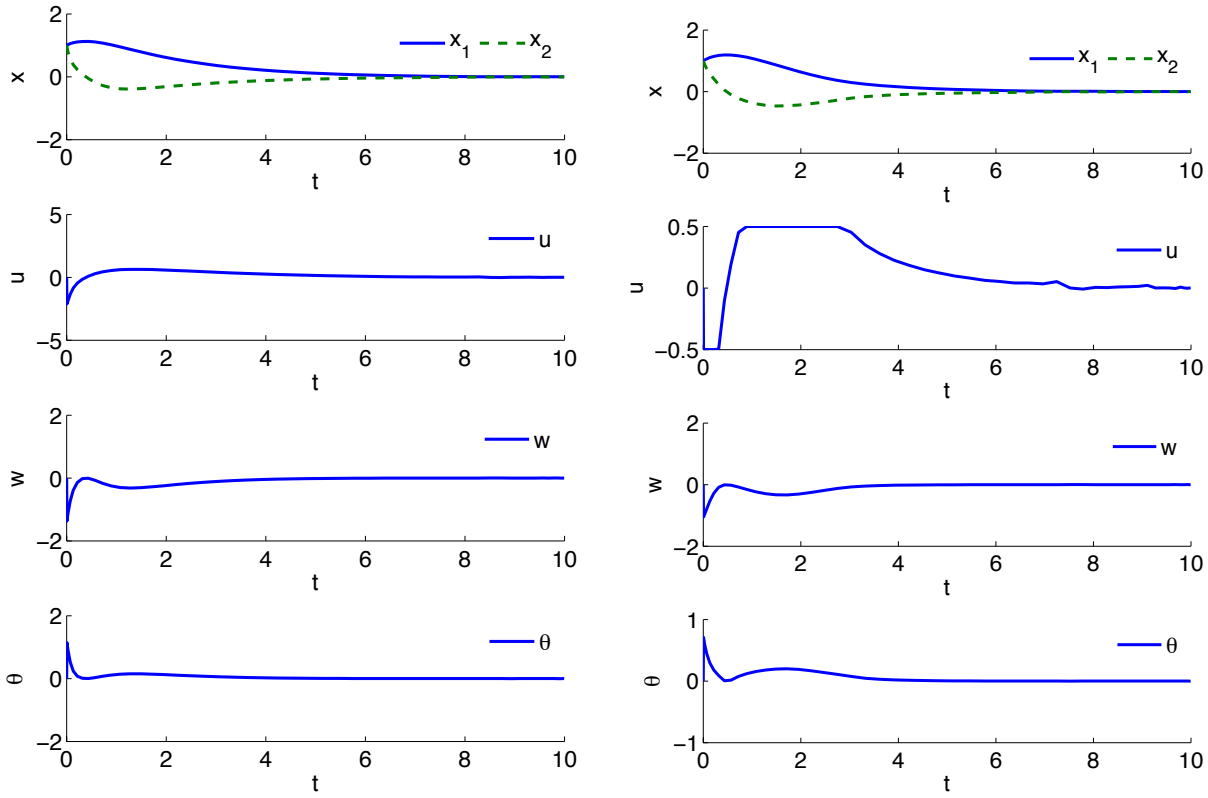
$$
\begin{aligned}
f(x) &= \begin{bmatrix} x_2(t) \\ -\frac{1}{2}x_2^5(t) - \frac{1}{2}x_2^3(t) - x_1(t) \end{bmatrix}, \quad g_1(x) = \begin{bmatrix} 0 \\ x_2(t) \end{bmatrix}, \quad g_2(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \\
H_2(x) &= \begin{bmatrix} 0 - x_2^2(t) \end{bmatrix} \\
E_1(x) &= x_2(t), \quad E_2(x) = 1, \quad h(x) = x_2(t), \quad F(x,\theta,t) = \theta(t)
\end{aligned} \tag{4.45}
$$

The uncertain term $\theta(t)$ can take on the form of any function (time-varying or not), assuming that the squared-norm of the function is bounded according to the assumptions given. An example is $\theta(t) = 0.01\sin(t)$, which can be interpreted as a small perturbation added to the nominal system parameter. We pursue an open-loop solution according to the direct collocation approach described in Section 4.3 and rewrite the optimization objective according to the complementarity conditions described. The resulting problem is passed to the solver KNITRO with trajectories approximated using Legendre polynomials.

To test the effectiveness of the trajectories against bounded disturbances, we implement a receding-horizon control (RHC) similar to the setup described in [105]. We compute trajectories for the system, apply the obtained control inputs for a fixed duration, then compute new trajectories using the current states as initial conditions. The total simulation time is 10s with a horizon of 0.1s, with 50 collocation points for each computed trajectory. The computation time for each trajectory is about 15s.

Fig. 4.1(a) gives a snapshot of the RHC at the beginning of a horizon, which takes in the specified initial conditions and compute trajectories according to the objective function in (4.42). Compared to the known global solution from Fig. 3.3(a), the state trajectories from Fig. 4.1(a) exhibit a slightly longer settling time. This is a direct result of the local nature of the solution.

One advantage of the trajectory optimization approach is the ability to consider con-



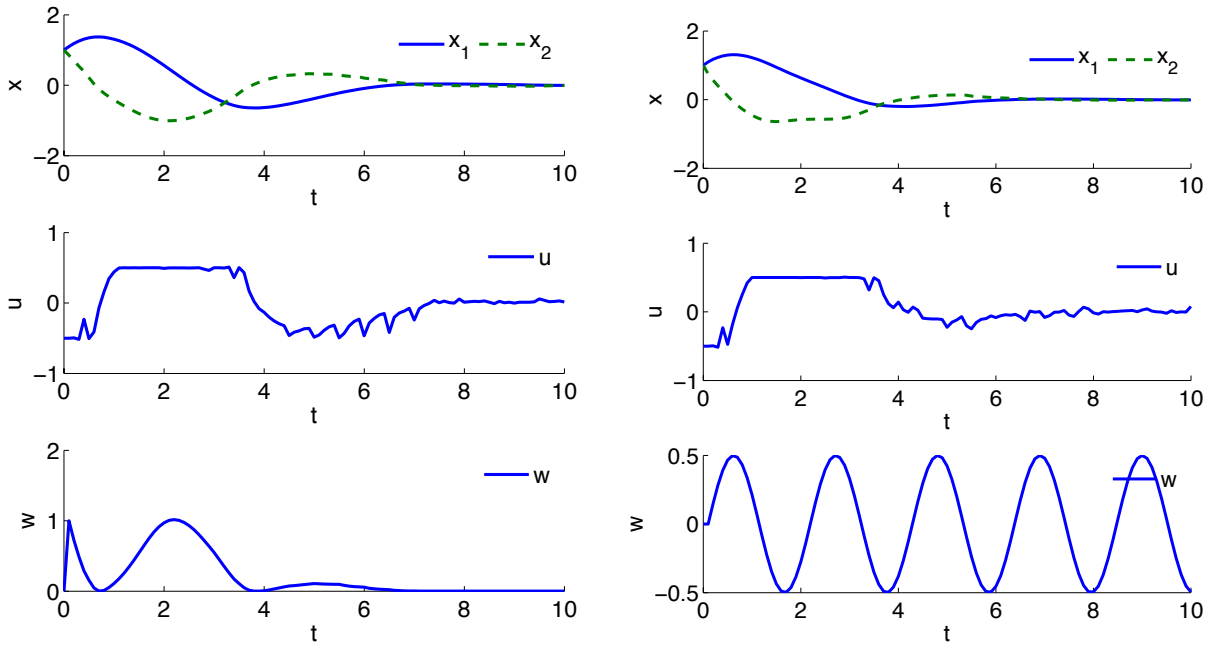
(a) Unconstrained input

(b) Constrained input to ±0.5

Figure 4.1: Optimized trajectories for a two-dimensional system under disturbance. Given constraints on the control input, stabilizing trajectories can still be obtained.

straints on the states and control inputs. In Fig. 4.1(b), we simulate a scenario where the control input was constrained to be ±0.5. The resulting control trajectory briefly saturated at ±0.5, but was still able to stabilize the system.

Fig. 4.2 shows a simulation of the RHC where the system is disturbed using the optimal disturbance computed in (3.58). Under the optimal disturbance, the settling time for the states are longer. Although the computed trajectories only estimated suboptimal disturbances, in a closed-loop setting, the RHC was able to stabilize the system with a



(a) Closed-loop response with optimal disturbance.

(b) Closed-loop response with sinusoidal disturbance.

Figure 4.2: Receding-horizon control of a two-dimensional uncertain system. The closed-loop response is stable for both optimal and sinusoidal disturbances.

performance close to the globally optimal solution in Fig. 3.3(a).

In direct trajectory optimization, the initial guess often plays an important role in determining the quality of the resulting solutions. In this two-dimensional example, random initial guesses were provided to the solver and the resulting trajectories were feasible though only locally optimal. In Chapter 6, we will analyze the computational performance of this robust trajectory optimizer in the context of a more complex dynamic system and provide a detailed discussion on the quality of solutions and computation time. In addition, we will also provide comparisons to alternative methods, such as LQR and non-robust trajectory optimization in a receding-horizon control framework.

4.5 Summary

In this chapter, we extended direct trajectory optimization to nonlinear robust $\mathcal{H}_\infty$ control under bounded external disturbances and parametric uncertainties. We solved the resulting differential game with open-loop trajectories obtained in accordance with necessary conditions for optimality. The proposed framework transformed a minimax optimization problem into a minimization problem with complementarity conditions. The resulting problem was solved by direct collocation pseudospectral methods, which transcribed a continuous-time problem into an equivalent discrete form by parameterizing the state and control spaces using global polynomials and collocating the differential-algebraic equations using nodes obtained from a Gaussian quadrature. We applied the open-loop trajectories in a receding-horizon control setting, which was shown to be effective on a simple two-dimensional benchmark problem.

To our knowledge, this is is the first application of direct trajectory optimization to the robust $\mathcal{H}_\infty$ control problem. Compared to a naive nonlinear programming approach, the proposed method was able to generate trajectories orders of magnitude faster, which makes it suitable for large-scale systems. The trade-off is the quality of the solutions obtained: for computing the trajectories, we can only guarantee that it satisfies a set of first order optimality conditions, and not the overall stability of the control design. In the next chapter, we extend this trajectory optimization approach to consider multiple models, which removes some of the restrictive assumptions made on the forms of the system uncertainties.

5

Multi-Model Robust Control

5.1 Motivation

This chapter proposes a new method for solving the robust $\mathcal{H}_\infty$ control problem presented in Section 1.3. In Chapter 3 and Chapter 4, we developed global and local methods for solving the robust $\mathcal{H}_\infty$ control problem under a differential game formulation with strong assumptions on the form of the system uncertainties. We modeled the external disturbance as a perturbation player, aiming to maximize the disturbance under parametric uncertainties, which are structured with bounded norms. In reality, the structured and additive nature of these modeled uncertainties may not be able to account for all forms of uncertainties with potential impacts on the performance of the system.

As discussed in Section 1.3.2, *unmodeled dynamics* is a phenomenon where the effects of higher-order dynamics (such as actuator dynamics) have a significant impact on the response of the system. To compensate for unmodeled dynamics is a difficult problem that requires a more flexible representation of uncertainties in the system dynamics than what we have previously proposed. In this chapter, we develop a heuristic approach to robust control design based on the concept of *multiple models*, reviewed in Section 1.2.7. A multi-model design evaluates the performance of a given control law over a distribution of possible system dynamics, which allow the optimization to consider the expected cost over a range of possible outcomes and account for the maximum cost for the worst-case

scenario. We consider this multi-model design in the context of differential games that we have developed in the previous chapters, with the added assumption that both players now consider a distribution of models in formulating their respective strategies. This is a philosophy consistent with the $\mathcal{H}_\infty$ designs proposed in the previous chapters.

The consequence of the proposed approach is a method that is able to address all three forms of uncertainties discussed in Section 1.3.2: external disturbances, parametric uncertainties, and unmodeled dynamics. The proposed method is scalable to high-dimensional systems by building on the robust trajectory optimization techniques developed in Chapter 4, which we extend to compute trajectories that simultaneously satisfy optimality conditions for multiple models. The resulting open-loop trajectories can be utilized in a receding-horizon setting to enable closed-loop feedback control. We illustrate the proposed approach using a simple two-dimensional system and discuss scalability to more complex applications.

5.2 Formulation

In the previous chapters, we formulated a robust $\mathcal{H}_\infty$-control problem with the nonlinear dynamics

$$\begin{aligned} \dot{x}(t) &= \Big[f(x) + \Delta f(x,\theta,t)\Big] + g_1(x)w + \Big[g_2(x) + \Delta g_2(x,\theta,t)\Big]u \\ z(t) &= [h(x), u]^T \\ x(t_0) &= x_0, \end{aligned} \tag{5.1}$$

with $x \in \mathcal{X}$ as the *state* vector, $z \in \mathcal{Z}$ is the *output* vector, $u \in \mathcal{U}$ and $w \in \mathcal{W}$ are the minimizing and maximizing players, respectively, x_0 is a vector of *initial states*, and the matrices $f(x)$, $g(x)$, and $h(x)$ are the *system matrices* that are smooth vector fields not

explicitly parameterized by time. The parameter $\theta \in \Theta$ is the uncertain system parameter which belongs to the set

$$\Theta = \{\theta | 0 \leq \theta \leq \theta_u\} \tag{5.2}$$

that is bounded above by θ_u. Parametrized by θ, the functions Δf and Δg_2 are uncertain functions belonging to the set of *admissible uncertainties*, defined by

$$\begin{aligned} \Delta f(x, \theta, t) &= H_2(x) F(x, \theta, t) E_1(x) \\ \Delta g_2(x, \theta, t) &= g_2(x) F(x, \theta, t) E_2(x), \end{aligned} \tag{5.3}$$

for $\|F(x, \theta, t)\|^2 \leq \beta$, which intuitively, assumes that the uncertainties are bounded by a sphere. Using these assumptions, we are able to derive necessary and sufficient conditions for optimality in terms of Isaacs' equation (3.21) and explicit feedback control laws (3.19) using the value function. In an open-loop setting, this model can be solved using both indirect and direct trajectory optimization, described in Section 4.2 and Section 4.3.

In this chapter, we formulate robust control problems which cannot be modeled by (5.1). In particular, we aim to remove the restrictive assumptions on the structure of the uncertainties in (5.3). It is clear not all parametric uncertainties can be posed in an additive way to the dynamics, as represented in (5.1). As an example, consider the linear optimal control problem posed by Atkeson [127], with double-integrator dynamics

$$\dot{x} = Ax + Bu, \tag{5.4}$$

where the system matrices are given by

$$A = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} T^2/2 \\ T \end{bmatrix}, \tag{5.5}$$

with $T = 0.001$s. The objective function is

$$J = \min_{u \in \mathcal{U}} \frac{1}{2} \int_{t_0}^{t_f} (x^T Q x + u^T R u)\, dt, \tag{5.6}$$

with

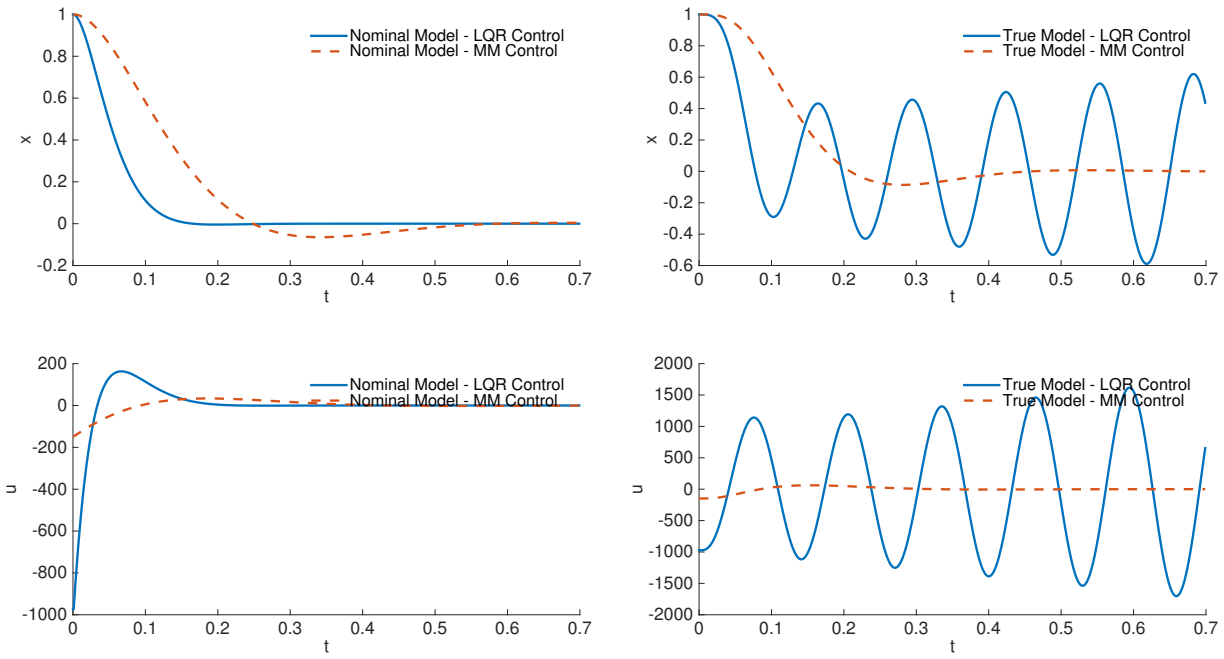
$$Q = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 0.001. \tag{5.7}$$

In addition to this nominal model, unmodeled dynamics include a second-order low pass filter with a cutoff frequency of 10Hz, under the dynamics

$$G(s) = \frac{\omega^2}{s^2 + 2\gamma\omega s + \omega^2} \tag{5.8}$$

with a damping ratio $\gamma = 1$ and a natural frequency $\omega = 20\pi$. There is no resonant peak and the unmodeled dynamics acts as a well-behaved low pass filter. As shown by Atkeson, when an LQR design is carried out using the nominal model without considering the unmodeled dynamics, the resulting controller failed to stabilize the overall system. A multi-model design for this system utilizes two models: a nominal model and an augmented model with an input filer with $\gamma = 1$ and $\omega = 10\pi$. The resulting controller is then able to stabilize both the nominal model and the true system model. Fig. 5.1 shows the closed-loop response for the LQR design and multi-model design for both the nominal model and true system model.

This example motivates our use of multiple models in control design. By evaluating the performance of a given control law over a distribution of possible system dynamics, we allow the optimization to consider the expected cost over a range of possible outcomes and account for the maximum cost for the worst-case scenario. This framework removes the need to to formulate uncertainties in the form (5.1) and allows flexibility in dealing with uncertainties both in the parameters of the dynamics and in the model structure.



(a) Nominal model

(b) True model with unmodeled dynamics

Figure 5.1: Comparison of LQR and multi-model control. LQR control is stable only for the nominal model, while a multi-model design is stable for both the nominal and the true model.

We propose to extend the multi-model framework in Atkeson [127] to nonlinear robust $\mathcal{H}_\infty$ control under a differential game formulation. In this setting, both players now consider a distribution of possible system dynamics in formulating their respective strategies, with the ability to compensate or exploit model uncertainties to optimize their respective objective functions. The resulting equilibrium is a *robust Nash equilibrium*, a pair of strategies (u^*, w^*) secured against any attempt by one player to unilaterally change his strategy, considering the underlying uncertainty.

The system dynamics we use in this multi-model design is given by

$$\begin{aligned} \dot{x}(t) &= f(x) + g_1(x)w + g_2(x)u \\ z(t) &= [h(x), u]^T \\ x(t_0) &= x_0. \end{aligned} \tag{5.9}$$

This is a simplified version of (5.1) that is a still an $\mathcal{H}_\infty$ formulation with an external disturbance term w, but without modeling the parametric uncertainties. We make the same assumptions about the boundedness of the disturbance, that is, the $\mathcal{L}_2$-gain

$$\int_{t_0}^{t_f} \|z(t)\|^2 \, dt \leq \gamma^2 \int_{t_0}^{t_f} \|w(t)\|^2 \, dt + \kappa(x_0), \quad \forall t_f > t_0, \tag{5.10}$$

is bounded by γ. To increase robustness against uncertainties, we instead define a family of such models.

Definition 5.1. (Multi-model system dynamics). The dynamics of the multi-model system is given by a set of first-order differential equations

$$\begin{aligned} \dot{x}^\alpha &= f^\alpha(x^\alpha) + g_1^\alpha(x^\alpha)w + g_2^\alpha(x^\alpha)u \\ z^\alpha &= [h^\alpha(x^\alpha), u]^T \\ x^\alpha(t_0) &= x_0^\alpha, \end{aligned} \tag{5.11}$$

where $\alpha \in \mathcal{A}$ belong to a finite parametric set $\mathcal{A}$. Here, x^α is the indexed state vector, u is the vector of shared controlled inputs to the set of systems, and w is the vector of shared external disturbances.

Each α represents a particular model from the parametric set with its own system matrices, which effectively increases the range of modeling options without requiring the uncertainty

terms to conform to (5.1). For the purposes of being consistent with previous chapters, we restrict the dynamics to be affine with respect to the control inputs.

We also augment the objective function based on this distribution of models.

Definition 5.2. (Multi-model objective function). The multi-model optimization objective takes the form

$$J(u,w) = \min_{u\in\mathcal{U}} \max_{\substack{w\in\mathcal{W}\\ \alpha\in\mathcal{A}}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\right]\, dt, \tag{5.12}$$

given a scalar γ and a time horizon $t_f > t_0$, over a distribution of models $\alpha \in \mathcal{A}$. We assume that $x(t) = 0$ is an unique equilibrium point of the system with $u(t) = 0$ and $w(t) = 0$.

Here, we minimize the worst-case cost of the family of dynamic models under disturbance. Clearly, this remains a minimax optimization problem, but now the control input u must stabilize a family of models, from which the worst-case cost is derived from.

5.3 Multi-model dynamic programming

Let us first consider the interpretation of the optimization objective in (5.12) from the view of the dynamic programming principle. Similar to the definition of the value function in (3.5), we can define a value function for cost (5.12) associated with each model in (5.11).

Definition 5.3. (Multi-model Bellman value function). The multi-model value function $V(x^\alpha, t)$ defines the cost-to-go from any initial state x^α and any time t such that

$$V^\alpha(x^\alpha, t) = \inf_{u\in\mathcal{U}} \sup_{w\in\mathcal{W},\alpha\in\mathcal{A}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z^\alpha(\tau)\|^2 - \gamma^2\|w(\tau)\|^2\right]\, d\tau. \tag{5.13}$$

In other words, the value function bounds the cost and provides the optimal cost-to-go for any state $x^\alpha(t)$. Given a family of such value functions $V^\alpha(x^\alpha, t)$, we can generate control laws for the two players by accounting for the corresponding state at each individual value function. The control laws take the form

$$\begin{aligned} u(t) &= \arg\min_{u\in\mathcal{U}} \left\{ |A|\Big(\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big) + \sum_{\alpha=1}^{|A|} V^\alpha(x^\alpha, t) \right\} \\ w(t) &= \arg\max_{w\in\mathcal{W}} \left\{ |A|\Big(\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big) + \sum_{\alpha=1}^{|A|} V^\alpha(x^\alpha, t) \right\}, \end{aligned} \tag{5.14}$$

where $|\mathcal{A}|$ is the cardinality of $\mathcal{A}$. This is the approach taken in Whitman [125] for multi-model optimal control problems and we have extended the definition of the value function to a two-player setting. Given that we have changed the system dynamics (5.1) into form (5.11), the necessary and sufficient conditions we have derived in terms of the Isaacs equations (3.21) no longer applies to this system. Thus, we must first derive a new Isaacs equation specific to the system dynamics in (5.11).

We first define the equilibrium strategies in the context of (5.11). For each model $\alpha \in \mathcal{A}$, the interpretation of the game from the perspective of both players are given by the objective functions

$$J_1(t, x^\alpha, u, w) = \min_{u\in\mathcal{U}} \frac{1}{2} \int_{t_0}^{t_f} \Big[\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big]\, dt \tag{5.15}$$

$$J_2(t, x^\alpha, u, w) = \max_{w\in\mathcal{W}} \frac{1}{2} \int_{t_0}^{t_f} \Big[\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big]\, dt. \tag{5.16}$$

These opposing objectives are related by

$$J_1(t, x^\alpha, u, w) = -J_2(t, x^\alpha, u, w), \tag{5.17}$$

which remains a noncooperative zero-sum game. A pair of strategies (u^*, w^*) forms a *robust Nash equilibrium* if

$$J_1(t, x^\alpha, u^*, w^*) \leq J_1(t, x^\alpha, u, w^*), \quad \forall u \in \mathcal{U}, \alpha \in \mathcal{A} \tag{5.18}$$

$$J_2(t, x^\alpha, u^*, w^*) \geq J_2(t, x^\alpha, u^*, w), \quad \forall w \in \mathcal{W}, \alpha \in \mathcal{A}. \tag{5.19}$$

Equivalently, the *saddle-point condition* indicates that

$$J(t, x^\alpha, u^*, w) \leq J(t, x^\alpha, u^*, w^*) \leq J(t, x^\alpha, u, w^*), \quad \forall w \in \mathcal{W}, u \in \mathcal{U}, \alpha \in \mathcal{A}. \tag{5.20}$$

We define the Bellman value function as a saddle-point equilibrium solution that satisfies

$$V^\alpha(x^\alpha, t) = \inf_{u \in \mathcal{U}} \sup_{w \in \mathcal{W}, \alpha \in \mathcal{A}} \frac{1}{2} \int_t^{t_f} \left[\|z^\alpha(\tau)\|^2 - \gamma^2 \|w(\tau)\|^2 \right] d\tau \tag{5.21}$$

and we use the notation

$$V_x^\alpha(x^\alpha, t) = \frac{\partial V(x^\alpha, t)}{\partial x^\alpha}, \quad V_t^\alpha(x^\alpha, t) = \frac{\partial V(x^\alpha, t)}{\partial t}, \tag{5.22}$$

to denote the row vectors of first partial derivatives of the value function with respect to t and x, respectively. For brevity, we drop the time index t when it is immaterial. We note that the value function bounds the cost function

$$V^\alpha(x^\alpha(t_f), t_f) \leq J^*(t, x^\alpha, u^*, w^*) \leq V^\alpha(t_0, x_0^\alpha), \tag{5.23}$$

where u^* and w^* are the optimal controls of the system (w^* is optimal in the sense that it provides the worst-case disturbance to the system).

Similar to Section 3.2, we use basic optimization theory to derive the optimal control and the associated necessary and sufficient condition for optimality. First, we convert

the constrained optimization problem in (5.12) to an unconstrained form by defining the appropriate Hamiltonian.

Definition 5.4. (Multi-model Hamiltonian). The multi-model Hamiltonian $H(x^\alpha, (V_x^\alpha)^T, u, w)$ for the system dynamics in (5.11) and objective function (5.12) is

$$H(x^\alpha, (V_x^\alpha)^T, u, w) = (V_x^\alpha)^T \Big(f(x) + g_1(x)w + g_2(x)u \Big) + \frac{1}{2}\|z^\alpha\|^2 - \frac{1}{2}\gamma^2\|w(t)\|^2, \quad (5.24)$$

where $(V_x^\alpha)^T$ is the *costate* for each model $\alpha \in \mathcal{A}$.

A necessary condition for optimality is Isaacs' condition, which states that

$$\inf_{u\in\mathcal{U}} \sup_{w\in\mathcal{W}} H(x^\alpha, (V_x^\alpha)^T, u, w) = \sup_{w\in\mathcal{W}} \inf_{u\in\mathcal{U}} H(x^\alpha, (V_x^\alpha)^T, u, w), \quad (5.25)$$

Equivalently, this can be expressed as

$$H(x^\alpha, (V_x^\alpha)^T, u^*, w) \leq H(x^\alpha, (V_x^\alpha)^T, u^*, w^*) \leq H(x^\alpha, (V_x^\alpha)^T, u, w^*). \quad (5.26)$$

For the Hamiltonian to be minimized over $\mathcal{U}$, the set of all possible controls, we write

$$\min_{\mathcal{U}} \sup_{\mathcal{W}} H(x^\alpha, (V_x^\alpha)^T, u, w) = 0. \quad (5.27)$$

The necessary condition for optimality is written as the partial derivatives of the Hamiltonian with respect to the control inputs

$$\begin{aligned} \frac{\partial H}{\partial u}(u^*, w) &= 0 \\ \frac{\partial H}{\partial w}(u, w^*) &= 0. \end{aligned} \quad (5.28)$$

These conditions give the control inputs with respect to the costate variable as

$$u^*(x, V_x^\alpha) = -g_2^T(x)(V_x^\alpha)$$
$$w^*(x, V_x^\alpha) = \frac{1}{\gamma^2} g_1^T(x)(V_x^\alpha). \tag{5.29}$$

Substituting back, we obtain

Theorem 5.5. (Necessary and sufficient condition for optimality). The multi-model Hamilton-Jacobi-Isaacs equation

$$V_t^\alpha(x^\alpha, t) + V_x^\alpha(x^\alpha, t) f^\alpha(x^\alpha) + \frac{1}{2} V_x^\alpha(x^\alpha, t) \left[\frac{1}{\gamma^2} g_1^\alpha(x)(g_1^\alpha)^T(x^\alpha) - g_2^\alpha(x^\alpha)(g_2^\alpha)^T(x^\alpha) \right] (V_x^\alpha)^T(x^\alpha, t) + \frac{1}{2}(h_1^\alpha)^T h_1^\alpha(x^\alpha) = 0, \tag{5.30}$$

is a necessary and sufficient condition for optimality for the multi-model optimization problem (5.12), subject to the system dynamics (5.11).

We omit the proof here since by construction, it satisfies the first-order necessary conditions (in (5.27)) and the sufficiency condition can be proven the same way as the single-model version of the Isaacs equation given in (3.21).

In the context of the multi-model optimization problem in (5.12), we have effectively reduced the problem to solving a series of Isaacs equations in (5.30), then computing the optimal feedback control laws in (5.14). In each case, the value function approximation approach described in Section 3.3 is fully applicable. Each value function can be represented in the form

$$\tilde{V}^\alpha(x^\alpha) = \sum_{i=1}^{n} w_i \phi_i(x^\alpha), \tag{5.31}$$

and the feedback control laws are given by

$$u = \arg\min_{u\in\mathcal{U}} \left\{ |A|\Big(\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big) + \sum_{\alpha=1}^{|A|} \tilde{V}^\alpha(x^\alpha, t) \right\}$$
$$w = \arg\max_{w\in\mathcal{W}} \left\{ |A|\Big(\|z^\alpha(t)\|^2 - \gamma^2\|w(t)\|^2\Big) + \sum_{\alpha=1}^{|A|} \tilde{V}^\alpha(x^\alpha, t) \right\}. \tag{5.32}$$

This is philospically similar to the approach presented by Whitman [125]. While this is feasible for simple systems with reduced dimensionality, it is far too impractical for large-scale systems. Instead of solving a series of these expensive optimization problems, we instead consider a new approach using trajectory optimization, which solves a single optimization problem in computing open-loop trajectories to (5.12).

5.4 Multi-model trajectory optimization

In Chapter 4, we presented a trajectory optimization approach for the nonlinear $\mathcal{H}_\infty$ control problem under the differential game formulation, where we computed open-loop trajectories that satisfied certain optimality conditions. In the context of a two-player differential game, the notation of open-loop solution entails that both players formulate their strategy at the moment the system starts to evolve, based on information available at the time: the system dynamics, the objective function, and initial conditions. This strategy cannot be changed once the system evolves and the saddle-point solution is the combination of strategies of both players which are secured against any attempt by one player to unilaterally change his strategy. We can extend this notion to the multi-model design, where both players now formulate open-loop solutions considering a distribution of possible system dynamics.

Consider again the distribution of dynamic models in (5.11) where

$$\begin{aligned} \dot{x}^\alpha &= f^\alpha(x^\alpha) + g_1^\alpha(x^\alpha)w + g_2^\alpha(x^\alpha)u \\ z^\alpha &= [h^\alpha(x^\alpha), u]^T \\ x^\alpha(t_0) &= x_0^\alpha. \end{aligned} \tag{5.33}$$

To formulate a trajectory optimization approach, we first concatenate the state vectors x^α and z^α for the individual models into a single vector in the form

$$\begin{aligned} X_\alpha &= \left(x^{\alpha_1}, \ldots, x^{\alpha_{|A|}}\right), \quad t \in [t_0, t_f] \\ Z_\alpha &= \left(z^{\alpha_1}, \ldots, z^{\alpha_{|A|}}\right), \quad t \in [t_0, t_f] \end{aligned} \tag{5.34}$$

and a concatenated vector field as a function

$$F_\alpha(X, u, w) = \left(f^{\alpha_1,u}(x^\alpha, u, w), \ldots, f^{\alpha_{|A|},u}(x^\alpha, u, w)\right). \tag{5.35}$$

We note that while the states and the associated dynamics have been combined to form a large system, the control inputs remain the same. In other words, there is a single control for the minimizing player u and a single control for the maximizing player w with simultaneously affects all models in $\mathcal{A}$. The objective function given this concatenated dynamics is

$$J(u, w) = \min_{u \in \mathcal{U}} \max_{w \in \mathcal{W}} \frac{1}{2} \int_{t_0}^{t_f} \left[\|Z_\alpha(t)\|^2 - \gamma^2 \|w(t)\|^2\right] \, dt, \tag{5.36}$$

subject to the dynamics

$$\dot{X}_\alpha = F_\alpha(X, u, w) \tag{5.37}$$

$$X(t_0) = X_0, \tag{5.38}$$

where X_0 is the vector of initial states for the concatenated states. We formulate an optimization problem to solve for u and w that are open-loop solutions that are functions of only time. In practical implementation, the obtained trajectories can be used without considering the concatenated state vector.

We formulate the trajectory optimization similarly to the approach described in Chapter 4 where we seek to transcribe a continuous time robust control problem into an equivalent mixed complementarity problem. Following the notation defined in Chapter 4, we redefine the independent variable to be $\tau : [\tau_0, \tau_f]$, which will be discretized to obtain the MCP. Rewriting the original objective function, we obtain

$$J = \Phi(X(\tau_0), X_\alpha(\tau_f), \tau_0, \tau_f) + \int_{\tau_0}^{\tau_f} C(X_\alpha(\tau), u(\tau), w(\tau))\, d\tau, \tag{5.39}$$

where $\Phi(X_\alpha(\tau_0), X_\alpha(\tau_f), \tau_0, \tau_f)$ is the terminal cost and $C(X_\alpha(\tau), u(\tau), w(\tau))$ is the one step cost in terms of the concatenated state vector X_α, while all other variables follow their original definitions. The dynamics in this discrete form is

$$\dot{X}_\alpha(\tau) = F_\alpha(X_\alpha(\tau), u(\tau), w(\tau)), \tag{5.40}$$

with the initial condition and final condition are expressed as boundary constraints

$$e(X_\alpha(\tau_0), X_\alpha(\tau_f), \tau_0, \tau_f) = 0. \tag{5.41}$$

We can also handle a mixture of constraints on the state and control variables, in the form of an inequality

$$s(X_\alpha(\tau), u(\tau), w(\tau)) \leq 0, \tag{5.42}$$

which is stacked to form the overall constraint set for the concatenated system. In this form, we can proceed with the trajectory optimizer described in Section 4.3.

5.5 Example - two-dimensional multi-model design

In the previous section, we assumed the system uncertainty to be a parametric time-varying function, which limits the form of uncertainty that can be modeled. Following the method described in Section 5.4, we augment the nominal model of the system

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{1}{2}x_2^5 - \frac{1}{2}x_2^3 - x_1 + x_2 w + u,\end{aligned} \tag{5.43}$$

by an additional two models

$$\begin{aligned}\dot{x}_3 &= x_4 \\ \dot{x}_4 &= -\left(\frac{1}{2} + \theta_1(t)\right) x_4^5 - \frac{1}{2}x_4^3 - x_3 + x_4 w + (1 + \theta_1(t))\, u\end{aligned} \tag{5.44}$$

and

$$\begin{aligned}\dot{x}_5 &= x_6 \\ \dot{x}_6 &= -\left(\frac{1}{2} + \theta_2(t)\right) x_6^5 - \frac{1}{2}x_6^3 - x_5 + x_6 w + (1 + \theta_2(t))\, u\end{aligned} \tag{5.45}$$

we assume that

$$\begin{aligned}\theta_1(t) &= 0.5 \sin(t + 5) \\ \theta_2(t) &= 0.5 \sin(t - 5).\end{aligned} \tag{5.46}$$

These are two different forms of parametric uncertainties, in addition to the nominal model, which forms a six-state system, from which we solve for a single control input u and a disturbance input w that are applied to all three models. To test the effectiveness

of the trajectories against bounded disturbances, we implement a receding-horizon control (RHC) similar to the setup described in Chapter 4. The total simulation time is 10s with a horizon of 0.1s, with 50 collocation points for each computed trajectory. The computation time for each trajectory is about 25s.

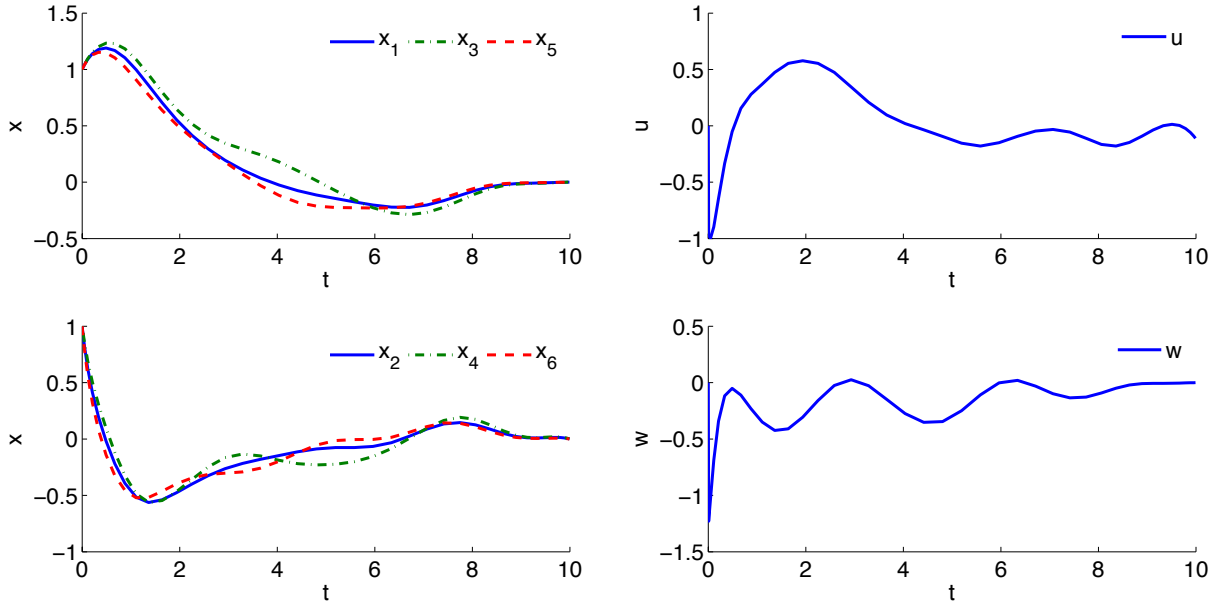
A snapshot of the trajectories at the beginning of the horizon is shown in Fig. 5.2. The x_1 and x_2 trajectories for all three models are compared in Fig. 5.2(a), and we see that there are visible differences in the three models. The input and disturbance trajectories are shown in Fig. 5.2(b), which the three models share. Similar to the results in Chapter 4, we can also apply constraints on the control inputs. Fig. 5.3(b) shows control trajectories generated, with saturations at $\pm = 0.5$. We see that despite the saturation, the resulting trajectories are stable.

So far, this three-model system represents system parametric uncertainty similar to the results obtained in Chapter 4. With a multi-model design, we can also address potential unmodeled dynamics. Consider the example raised in (5.8), where a second-order low pass filter with a cutoff frequency of 10Hz is applied to the control input u. In state space form, this low pass filter is equivalent to a linear system

$$\begin{aligned} \dot{x}_f &= A_f x_f + B_f u \\ u_f &= C_f x_f + D_f u, \end{aligned} \tag{5.47}$$

where the input u is low-pass filtered to produce the new input u_f. The system matrices are given by

$$\begin{aligned} A_f &= \begin{bmatrix} 1.9112 & -0.9150 \\ 1 & 0 \end{bmatrix}, \qquad B_f = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ C_f &= \begin{bmatrix} 0.0037, 0.0001 \end{bmatrix}, \qquad D_f = 9.4469\text{E-}04. \end{aligned} \tag{5.48}$$



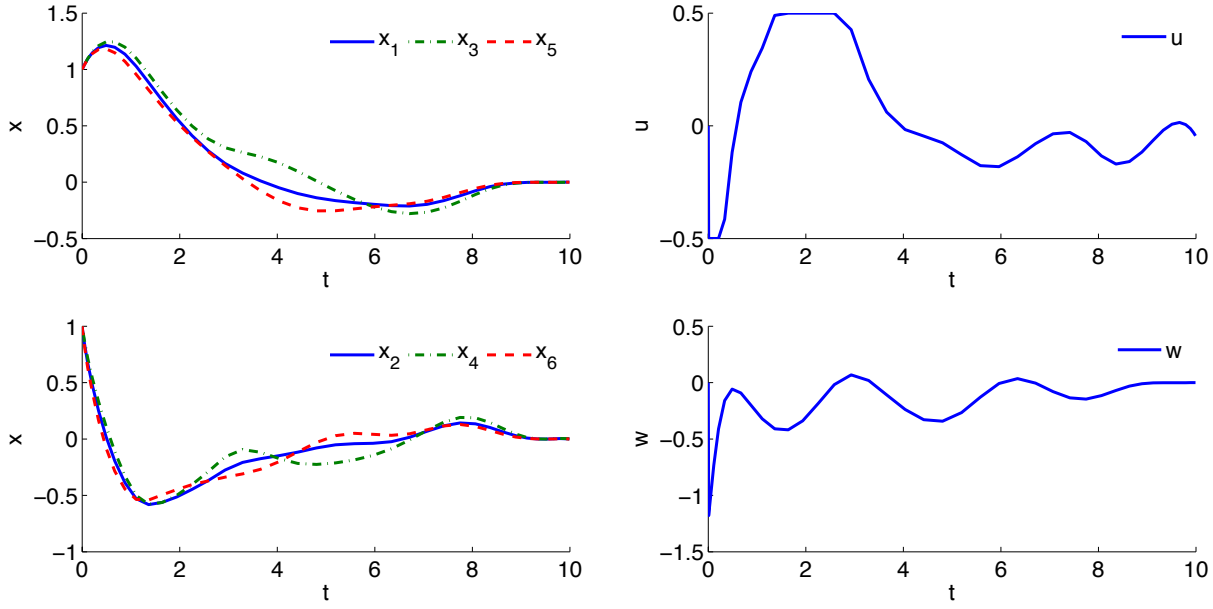
(a) State trajectories

(b) Input and disturbance trajectories

Figure 5.2: Optimized trajectories for a multi-model two-dimensional system under disturbance. Two models with varying parameters are provided in addition to the nominal model under identical disturbance, and a single control input stabilizes all three systems.

The overall system is given by two model. First, the nominal model

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{1}{2}x_2^5 - \frac{1}{2}x_2^3 - x_1 + x_2 w + u \end{aligned} \tag{5.49}$$



(a) State trajectories

(b) Input and disturbance trajectories

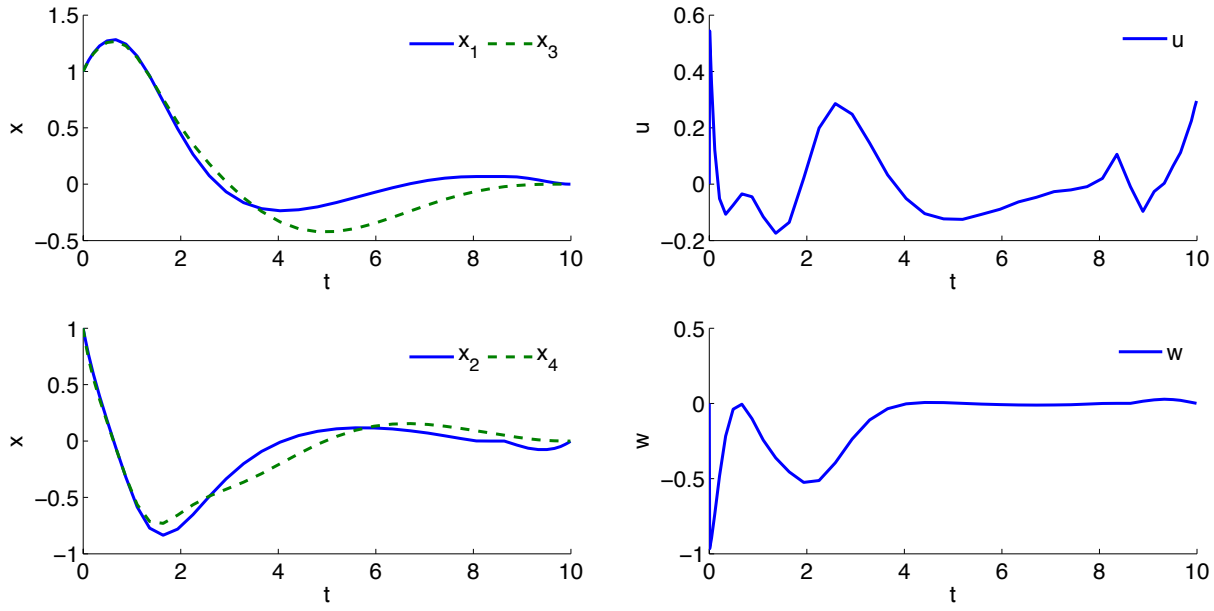
Figure 5.3: Optimized trajectories for a multi-model two-dimensional system under disturbance with control input constraints. Two models with varying parameters are provided in addition to the nominal model under identical disturbance, and a single constrained control input (± 0.5) stabilizes all three systems.

and the additional model

$$\begin{aligned}
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -\frac{1}{2}x_2^5 - \frac{1}{2}x_2^3 - x_1 + x_2 w + (C_f x_f + D_f u) \\
\dot{x}_f &= A_f x_f + B_f u.
\end{aligned} \tag{5.50}$$

In total, the concatenated state space of the system consists of six states - two for the nominal model and four for the additional model, which includes two states for the filter.

We can now apply this new multi-model design in a receding-horizon setting with the



(a) State trajectories

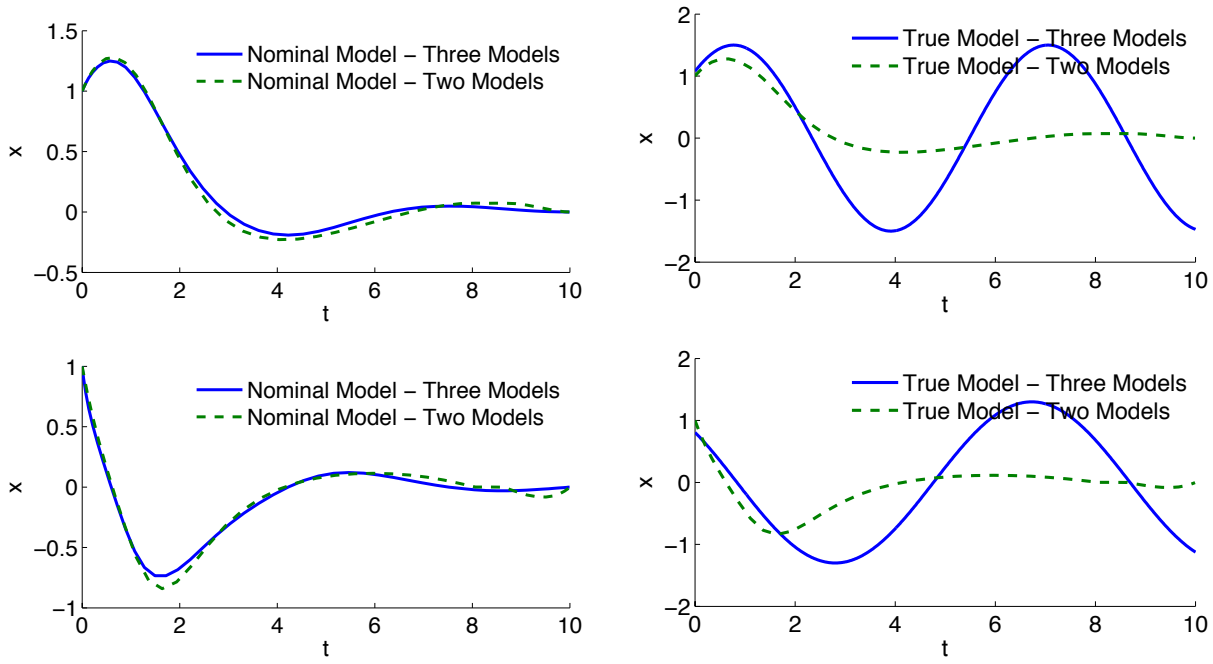
(b) Input and disturbance trajectories

Figure 5.4: Optimized trajectories for a multi-model two-dimensional system under disturbance. In addition to the nominal model for the system dynamics, an additional model accounts for unmodeled dynamics in the form of a low-pass filter. A single control trajectory stabilizes both systems under identical disturbance.

same parameter as previously described. Fig. 5.4 shows a snapshot of the trajectories obtained for this system in the beginning of a horizon.

We can now examine the performance of the two RHC setups described. The first RHC setup consists of three models which only model parametric uncertainty, described in (5.46). The second RHC setup consists of two models, which accounts for unmodeled dynamics in the form of a low-pass filter, described in (5.47). We test the RHC using both the nominal model and the true model which includes the unmodeled dynamics. In both cases, we assume the actual parametric uncertainty to be a sinusoidal signal $\theta(t) = 0.01\sin(t)$.

Fig. 5.5 shows a simulation of the RHC where the system is disturbed using the optimal



(a) Nominal model

(b) True model with unmodeled dynamics

Figure 5.5: Comparison of two robust RHC setups. RHC using three models, accounting only for parametric uncertainty, is stable only for the nominal model. A two-model design accounting for unmodeled dynamics is stable for both the nominal and the true model.

disturbance computed in (3.58). As shown in Fig. 5.5(a), the first RHC with three models exhibits better performance with respect to the nominal model, while the second RHC is overly conservative, leading to a longer settling time for both states. However, with respect to the true model, Fig. 5.5(b) shows that the three-model RHC failed to stabilize the system.

In Chapter 6, we will analyze the computational performance of this multi-model trajectory optimizer in the context of a more complex dynamic system and provide a detailed discussion on the quality of solutions and computation time. In addition, we will also provide comparisons to alternative methods, such as LQR and non-robust trajectory opti-

mization in a receding-horizon control framework.

5.6 Summary

In this chapter, we developed a heuristic approach to robust control design based on the concept of multiple models, which evaluated the performance of a given control law over a distribution of possible system dynamics. A distribution of models allowed the optimization to consider the expected cost over a range of possible outcomes and account for the maximum cost for the worst-case scenario. We improved on the approaches presented in the previous chapters by removing restrictive assumptions made on the forms of uncertainties, which enabled the approach to compensate for effects such as unmodeled dynamics. We developed these results in the context of necessary sufficient conditions for individual value functions similar to Chapter 3 and utilized the robust trajectory optimization approach in Chapter 4 to compute open-loop trajectories that simultaneously satisfied the optimality conditions for multiple models. We demonstrated the use of these trajectories in a receding-horizon setting for a simple two-state system.

The proposed multi-model design is a powerful approach to differential games. Where we have proposed a distribution of system dynamics, the approach can be easily extended to scenarios where a distribution of different objective functions and information structures are given to the players, resulting in an even-wider discrepancies on the knowledge and belief about the underlying game. This could potentially be used to model respective decision makers in a large scale systems, for example, in a mean-field games setting.

6 Results

6.1 Overview

This chapter presents experimental validation of the nonlinear robust control designs presented in Chapter 3, Chapter 4, and Chapter 5. We base our results on three platforms: a three-link planar model of a bipedal humanoid robot, a seven degrees-of-freedom robot manipulator, and a 15 degrees-of-freedom model based on the Atlas bipedal humanoid robot from Boston Dynamics. These models are chosen for their varying degrees of complexity and we test the proposed nonlinear robust control designs on these models using different combinations of initial conditions, parameters, and forms of uncertainties. In these results, we hope to experimentally demonstrate the significant effects of various forms of uncertainties on the behaviors of the closed-loop system and hence the need to robustify standard controllers. We also discuss the potential advantages and disadvantages of the techniques proposed in the previous chapters.

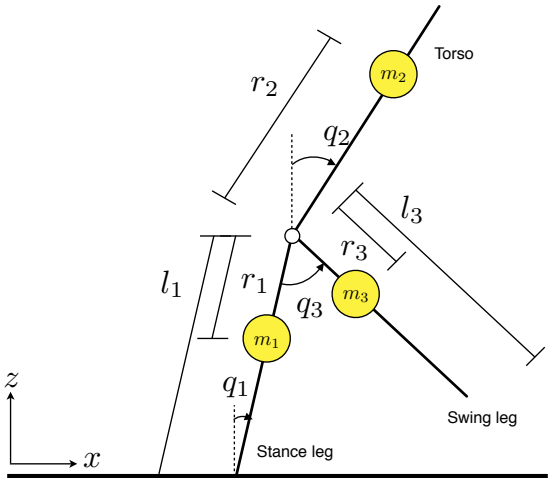


Figure 6.1: Planar model of a bipedal humanoid robot with three links. The three links are referred to as the stance leg (l_1), torso (l_2), and swing leg (l_3), respectively. Of the three joints, q_1 is stance ankle angle, q_2 is the angle of the torso with respect to the vertical, and q_3 is the angle of the swing leg with respect to the stance leg. The parameters for the model are given in Table 6.1.

6.2 Three-link planar bipedal humanoid model

6.2.1 Model

We model a three-link bipedal humanoid robot in the sagittal plane. A diagram for the system is shown in Fig. 6.1, with the physical parameters listed in Table 6.1. Of the three joints, q_1 is stance ankle angle, q_2 is the angle of the torso with respect to the vertical, and q_3 is the angle of the swing leg with respect to the stance leg. The three links are referred to as the stance leg (l_1), torso (l_2), and swing leg (l_3), respectively, and we follow a standard Lagrangian approach to derive the rigid-body dynamics for the model. We define the generalized coordinates of the system $q(t) = (q_1(t), q_2(t), q_3(t))^T$ according to Fig. 6.1

Table 6.1: Physical parameters for a planar three-link bipedal model

	m (kg)	r (m)	I_y $(kg \cdot m^2)$	l (m)
Stance leg (l_1)	10.9	0.37	0.57	0.77
Torso (l_2)	65	0.4	4.3	-
Swing leg (l_3)	12.6	0.43	0.85	0.81

and the *Lagrangian*

$$\mathcal{L} = \sum_i k_i - \sum_i p_i \tag{6.1}$$

is formed by taking the difference between the sum of kinetic energies and the sum of potential energies. The *Euler-Lagrange equation* is

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = \tau_i, \tag{6.2}$$

where τ_i is the external force acting on the ith generalized coordinate, which in this case, is individual actuated degree of freedom in the robot. Solving the Euler-Lagrange equation gives the nonlinear equations of motion, which can be organized in a form that is linear in acceleration and torque

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau, \tag{6.3}$$

where M is the *mass matrix* that is square, symmetric, and positive definite; C consists of centripetal and Coriolis forces; G represents the gravitational and other forces acting on the robot; and τ is a vector of torques that are the controlled inputs to the system. In addition, we note that $(\dot{M} - 2C)$ is a skew-symmetric matrix. This form is commonly referred to as the *manipulator equation* [135].

Rearranging the terms and inverting the mass matrix, we can rewrite (6.3) as a series

of first-order nonlinear equations

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ z &= [h(x), u]^T \\ x(0) &= x_0, \end{aligned} \tag{6.4}$$

with $x \in \mathcal{X}$ is the *state* vector, $z \in \mathcal{Z}$ is the *output* vector, $u \in \mathcal{U}$ is the vector of *controlled inputs* to the system, x_0 is a vector of *initial states*, and the matrices $f(x)$ and $g(x)$

$$\begin{aligned} f(x) &= -M^{-1}(q)\left(C(q,\dot{q})\dot{q} + G(q)\right) \\ g(x) &= M^{-1}(q), \end{aligned} \tag{6.5}$$

are the *system matrices* that are smooth vector fields. Following the uncertainty and disturbance modeling described in Section 1.3.2, we augment the system with uncertain terms $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$, as well as the external disturbances w_1, w_2, and w_3 to the three joints. We use w to denote a vector of external disturbances with the associated system matrix $g_1(x)$. For the parametric system uncertainty, we use the term θ to represent an offset to the mass of the torso link, with the nominal value being 65kg, as listed in Table 6.1.

The final description of the problem is

$$\begin{aligned} \dot{x}(t) &= [f(x) + \Delta f(x,\theta,t)] + g_1(x)w(t) + [g_2(x) + \Delta g_2(x,\theta,t)]\,u(t) \\ z(t) &= [h_1(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{6.6}$$

where $f(x)$, $g_1(x)$, $g_2(x), h(x)$ are system matrices and $\Delta f(x,\theta,t)$ and $\Delta g_2(x,\theta,t)$ are

parameterized uncertain terms. We use the same optimization objective

$$J(x,u,w,\theta) = \min_{u\in\mathcal{U}} \max_{\substack{w\in\mathcal{W}\\ \theta\in\Theta}} \frac{1}{2}\int_{t_0}^{t_f} \left[\|z(t)\|^2 - \gamma^2\|w(t)\|^2\right] dt \tag{6.7}$$

as stated in the previous chapters.

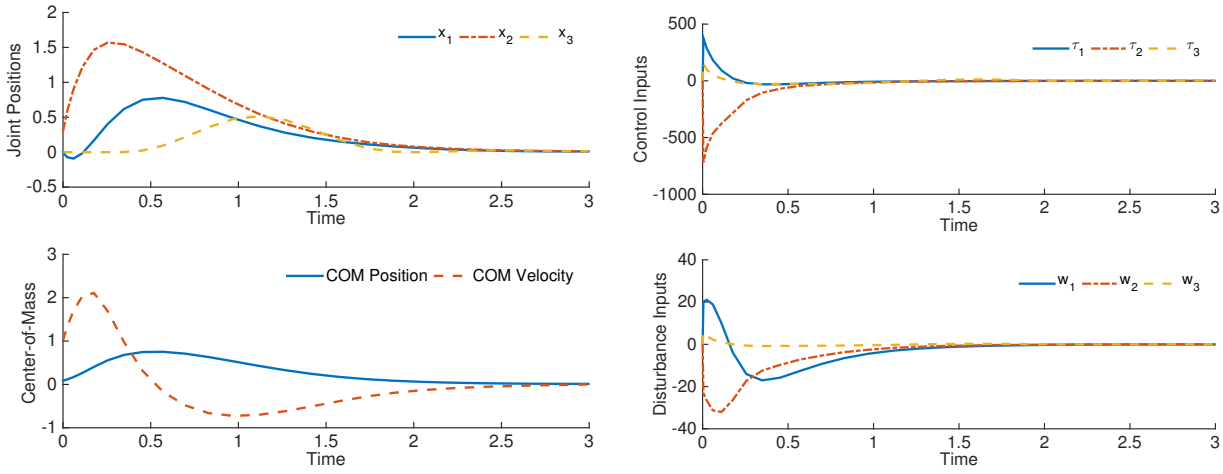
6.2.2 Standing balance control

Due to the nonlinear nature of the dynamics, no analytical solutions exist for this model. Although the dimensionality of this planar three-link model is relatively low compared to other problems investigated in this chapter, the computational burdens of pursuing a numerical dynamic programming-based solution, as described in Section 3.2, remains high and would require extensive computational machinery for a parallelized implementation. Instead, we implemented a value function approximation based control design, as described in Section 3.3. This is done to obtain insight into the performance of the proposed approximation scheme in Section 3.4.3, as well as to establish a benchmark for the single and multiple model trajectory optimization.

Given the approximate value function $\tilde{V}(x)$ as a weighted sum of a set of basis functions ϕ_i

$$\tilde{V}(x) = \sum_{i=1}^{n} w_i \phi_i(x), \tag{6.8}$$

with $\phi_i(x)$ given by Fourier basis functions. In this case, the system of integral equations must be solved numerically, with a numerical quadrature embedded within a trust region dogleg iteration to solve the system of equations. To approximate the value function, we used the Fourier-basis functions described in Section 3.3 in an uncoupled setting. As described in Section 3.3, for an k-th order Fourier approximation in n-dimensions, there are $2(k+1)^n$ basis functions, which is an exponential explosion of parameters in the number



(a) State trajectories

(b) Input and disturbance trajectories

Figure 6.2: Control design based on value function approximation. A planar three-link model is stabilized under bounded external disturbance and internal parametric uncertainties.

of dimensions. Following the procedure to uncouple the basis functions, there are $n(k+1)$ number of uncoupled basis functions. With $k = 35$ and $n = 6$, this is the difference between $4,353,564,672$ coupled and 216 uncoupled basis functions.

We model the scenario of a three-link robot in the standing, upright position with initial velocities. Fig. 6.2 gives the closed-loop response of the value function approximation-based control design given a set of initial conditions under the nominal model. Fig. 6.2(a) shows the stable state trajectories along with the center-of-mass (COM) position and velocity in the horizontal direction. We see that due to the external disturbance, the COM position initially moves in the positive direction before stabilizing back to the upright position. Fig. 6.2(b) shows the corresponding control inputs and disturbance inputs.

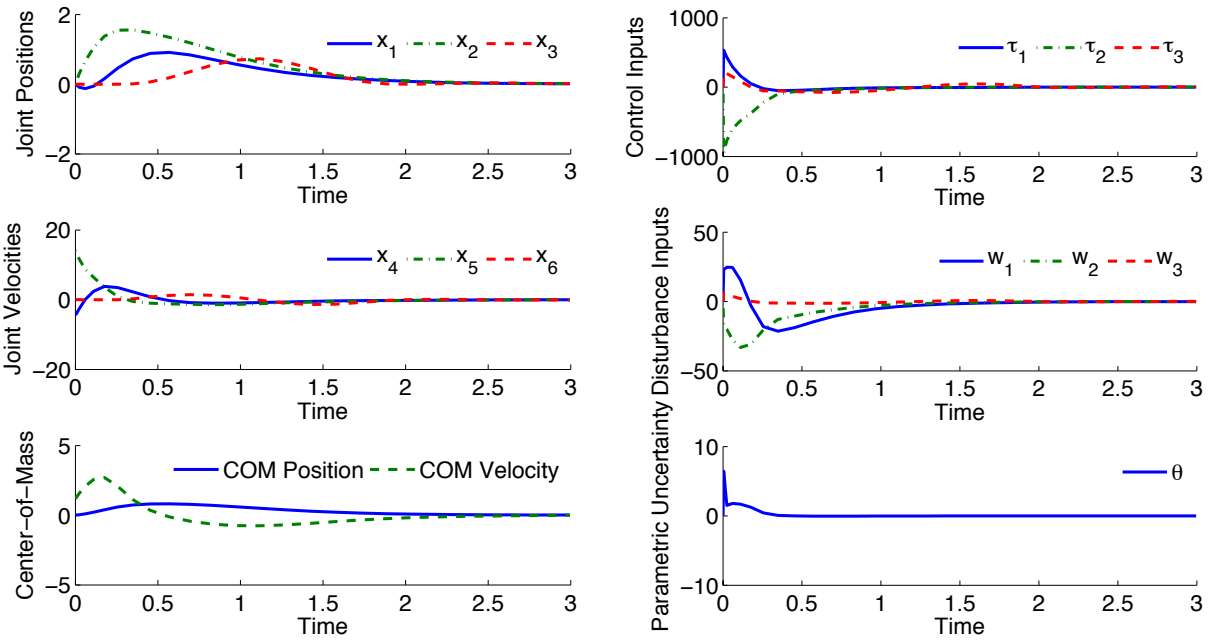
The global necessarily and sufficient conditions for optimality derived in Section 3.4.2 would have only limited utility in this model since it is difficult to incorporate state and input constraints to model joint and actuator limits. Doing so would require formulating

additional terms in the Hamiltonian to incorporate the constraints, which significantly complicates the implementation of the resulting control design. While other numerical implementations for dynamic programming can be used for this purpose (see for example, Section 1.2.4), we only include an unconstrained version of the optimization problem to establish a baseline performance for subsequent analysis. Due to the dimensionality of the problem, we also must make a simplifying assumption on the coupled nature of the value function. The trade-off of an uncoupled value function approximation is the quality of the solution with computational efficiency, and we discuss these issues more in-depth in Section 6.5.

The main focus of this section is to apply the single-model trajectory optimization from Chapter 4 and multi-model trajectory optimization in Chapter 5 under a receding-horizon control formulation. Since these techniques can easily incorporate additional constraints on the states and control inputs, we also simulate the effects of these constraints on the closed-loop performance. To demonstrate the performance of these robust trajectory optimizers, they are tested against a LQR controller designed with a linearized model of the dynamics, as well as a traditional non-robust trajectory optimizer that does not explicitly model system uncertainties.

For the RHC setups, we model a robot in the standing, upright position with initial velocities. For the three trajectory-based approaches (standard, single model robust, and multi-model robust), we implement a receding-horizon control (RHC) similar to the approach described in Chapter 4 and Chapter 5. We compute trajectories for the system, apply the obtained control inputs for a fixed duration, then compute new trajectories using the current states as initial conditions. For all three approaches, the total simulation time is 3s with a horizon of 0.1s, with 30 collocation points for each computed trajectory.

Fig. 6.3 gives a snapshot of the RHC at the beginning of a horizon, which takes in the specified initial conditions and computes trajectories based on the nominal model accord-



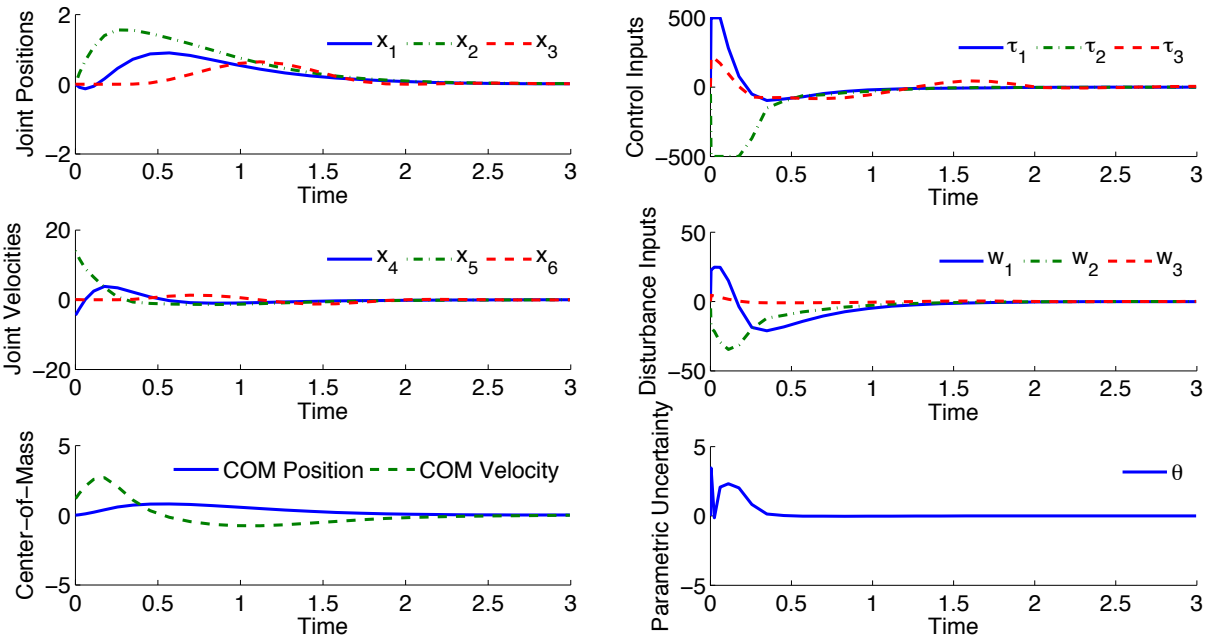
(a) State trajectories

(b) Input and disturbance trajectories

Figure 6.3: Snapshot of the receding-horizon control for a planar, bipedal robot with three links under disturbance. The optimized trajectories are generated using the nominal model at the beginning of the horizon with the specified initial conditions modeling the robot in a standing, upright position with some initial velocities. The stable trajectories maintaining the standing balance of the robot are applied for a fixed duration of the RHC horizon.

ing to the objective function in (6.7). These results are extremely similar to Fig. 6.2(a) using the value function approach with a near identical form of external disturbance. The COM position initially moves in the positive direction before stabilizing back to the upright position. We see that the worst-case internal parametric uncertainty is achieved by increasing the parameter θ, which is the mass of the torso link, as shown in Fig. 6.3(b).

Since the trajectory optimizer is able to handle constraints on the control inputs, we repeat the standing balance experiment but constrain the three control inputs u_1, u_2, and



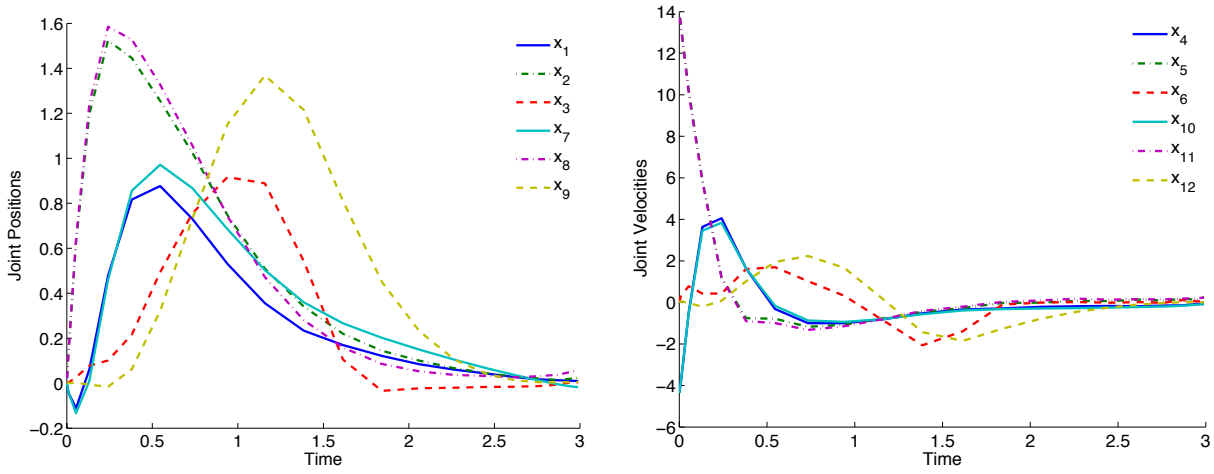
(a) State trajectories

(b) Input and disturbance trajectories

Figure 6.4: Snapshot of the receding-horizon control for a planar, bipedal robot with three links under disturbance. The trajectories are generated with a single model with constrained control inputs. The stable trajectories maintaining the standing balance of the robot despite limited control authority.

u_3 to ± 500. The resulting trajectories are given in Fig. 6.4, where there is a region of control saturation in τ_2 but the resulting state trajectories remain stable. Fig. 6.4(a) gives the state trajectories and COM position and velocity in the horizontal direction. Fig. 6.4(b) gives the corresponding control input and disturbance trajectories.

With a multi-model design, we can also address potential unmodeled dynamics. Consider the example raised in (5.8), where a second-order low pass filter with a cutoff frequency of 10Hz is applied to the control input u. In state space form, this low pass filter



(a) State trajectories

(b) Input and disturbance trajectories

Figure 6.5: Snapshot of the receding-horizon control for a planar, bipedal robot with three links under disturbance. The trajectories are generated with two-model design, with 18 concatenated states. The stable trajectories maintaining the standing balance of the robot for both models.

is equivalent to a linear system

$$\begin{aligned} \dot{x}_f &= A_f x_f + B_f u \\ \dot{u}_f &= C_f x_f + D_f u, \end{aligned} \tag{6.9}$$

where the input u is low-pass filtered to produce the new input u_f. Given three control inputs u_1, u_2, and u_3, we designed three low-pass filters according to (6.29) to produce the filtered control inputs. These filtered inputs are used in the dynamics for a second model to augment the nominal model. In total, this multi-model design contains 18 concatenated states: six for the nominal model, six for the additional model, and two states for each filter. As shown in Fig. 6.5(a), the addition of the unmodeled dynamics has a significant effect on the response of the system, as the states x_2 and x_3 have widely different joint

Table 6.2: Model uncertainties for a planar three-link bipedal humanoid

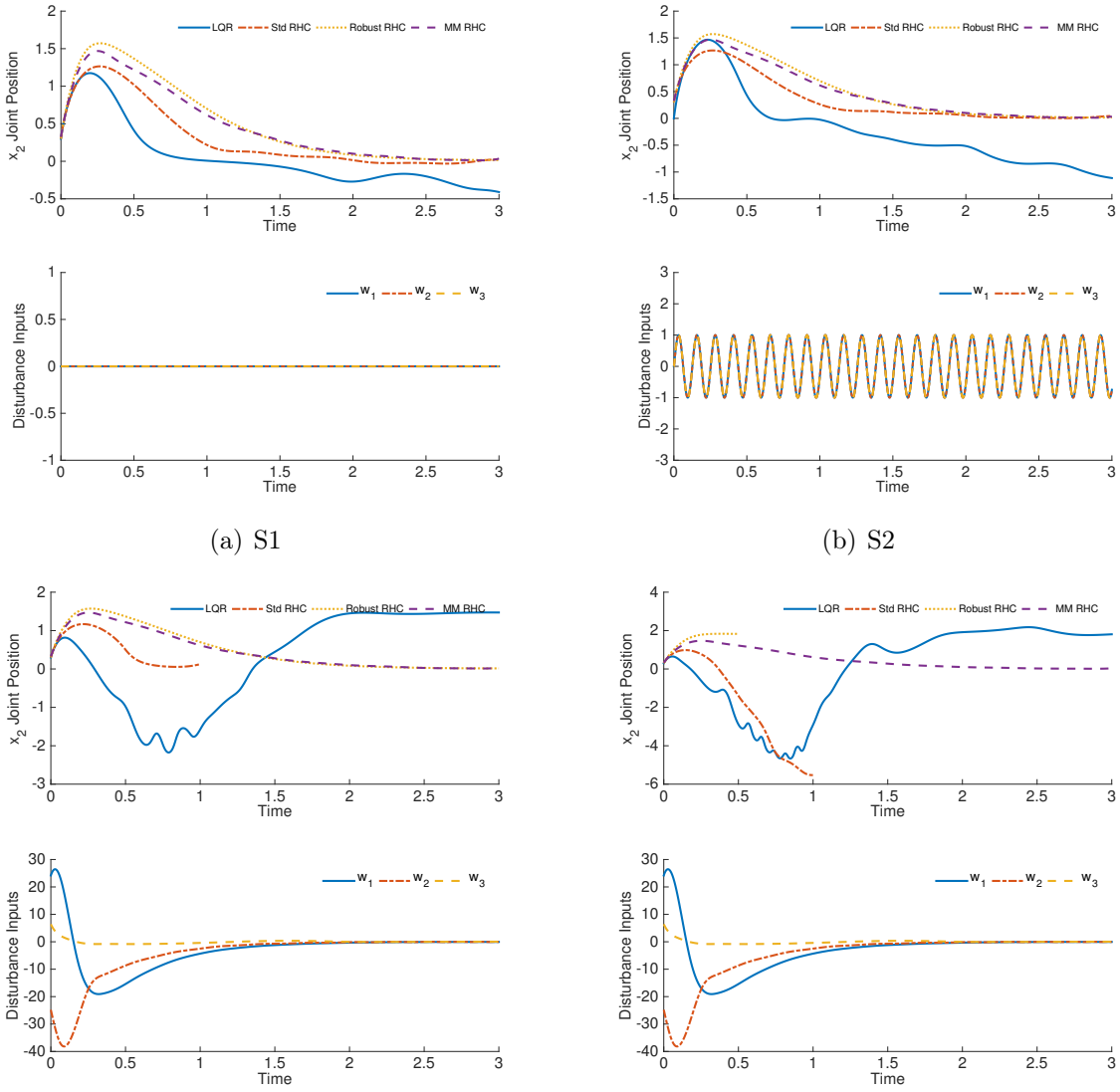
	S1	S2	S3	S4
External Disturbance	None	Sinusoidal	Open-loop	Open-loop
Parametric Uncertainty	None	None	Sinusoidal	Sinusoidal
Unmodeled Dynamics	None	None	None	Low-pass filter

trajectories between the nominal and the second model (with x_8 and x_9). Despite these differences, the control input stabilized all trajectories in the two models.

We now compare the single and multi-model robust trajectory optimization against an LQR controller designed with a linearized model of the dynamics, as well as a traditional non-robust trajectory optimizer that does not explicitly model system uncertainties. Four scenarios $S1$-$S4$ were set up using various combinations of initial conditions, external disturbances, parametric uncertainties, and unmodeled dynamics. Table 6.2 lists the conditions for the four scenarios.

In S1, the four approaches were tested with no external disturbances, no parametric uncertainties, and no unmodeled dynamics. The non-zero initial condition for the simulation was set up outside the linear domain of attraction of the LQR control. As expected, the LQR control failed to stabilize the system, while the three RHC-based approaches successfully stabilized the system, with similar closed-loop responses. The closed-loop response of the four approaches are shown in Fig. 6.6(a).

In S2, the four approaches were tested with sinusoidal external disturbances, no parametric uncertainties, and no unmodeled dynamics. The three links start from an upright position with zero initial conditions, but the external disturbances perturbed the system, causing LQR control to fail. The three RHC-based approaches continued to be able to stabilize the system given the external disturbances. The closed-loop response of the four approaches are shown in Fig. 6.6(b).



(a) S1

(b) S2

(c) S3

(d) S4

Figure 6.6: Comparison of LQR, standard RHC, robust single-model RHC, and robust multi-model RHC. Four scenarios $S1$-$S4$ were setup using various combinations of initial conditions, external disturbances, parametric uncertainties, and unmodeled dynamics listed in Table 6.2

In S3, the four approaches were tested with an open-loop external disturbance computed using the single-model robust trajectory optimization. In addition, we augment the system with a sinusoidal parametric uncertainty on the torso mass m_2, but no unmodeled dynamics. Given that this setup matches closely with the assumptions made by the single-model RHC, we expect that it will have the best performance. Indeed, the standard RHC failed to stabilize the system and the states of the system violated the constraints of the solver around $t = 1$s. The single and multi-model RHC both were able to stabilize the system, with similar performances. The closed-loop response of the four approaches are shown in Fig. 6.6(c).

In S4, the four approaches were tested with an open-loop external disturbance computed using the single-model robust trajectory optimization, sinusoidal parametric uncertainty, and unmodeled dynamics in the form of a low-pass filter on the control inputs. In this case, only the multi-model RHC was able to stabilize the system. The closed-loop response of the four approaches are shown in Fig. 6.6(d).

One important design parameter in addressing model uncertainty is the parameter γ in the optimization objective (6.7). As discussed in Section 3.2, γ bounds the disturbance inputs w and can be adjusted to vary the amount of disturbance given to the robot. To visualize the effect of γ, we generated center-of-mass (COM) position and velocity in the horizontal direction with respect to time, and plotted the phase diagram in Fig. 6.7 using several choices of γ. The trajectories show that an initial non-zero COM location can be stabilized back to the upright position, given disturbances of various magnitudes.

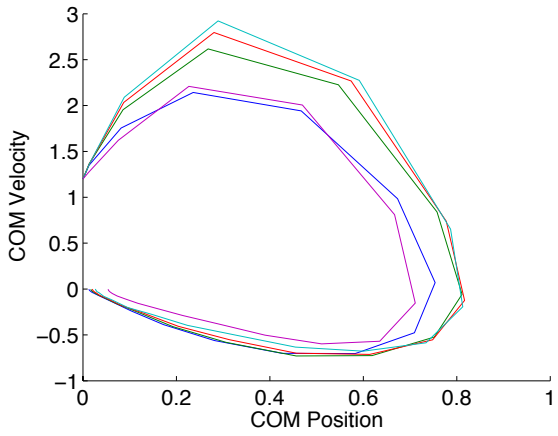


Figure 6.7: Phase diagram generated using center-of-mass (COM) position and velocity in the horizontal direction with respect to time. The trajectories show that an initial non-zero COM location can be stabilized back to the upright position, given disturbances of various magnitudes specified using γ. No parametric uncertainties are specified.

6.3 KUKA KR 5 sixx R650 manipulator control

6.3.1 Operational space control

In this section, we extend the proposed robust control designs to implement the operational space control of a robotic manipulator. As with Section 6.2, we assume the dynamics to take the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau, \tag{6.10}$$

where M is the *mass matrix* that is square, symmetric, and positive definite; C consists of centripetal and Coriolis forces; G represents the gravitational and other forces acting on the robot; and τ is a vector of torques that are the controlled inputs to the system. To be

consistent with the notations used in previous work, we also define

$$F(q, \dot{q}) = -C(q, \dot{q}) - G(q) \tag{6.11}$$

and hence the manipulator equation can be written in the form

$$M(q)\ddot{q} = u + F(q, \dot{q}). \tag{6.12}$$

We assume that the task description can be written in the form of a constraint given by the function

$$h(q, \dot{q}, t) = 0. \tag{6.13}$$

Particularly, for a class of tasks that can be formulated as

$$A(q, \dot{q}, t)\ddot{q} = b(q, \dot{q}, t), \tag{6.14}$$

which can be achieved for most tasks by differentiating $h(q, \dot{q}, t)$ with respect to time. This class of problems has an elegant optimization-based solution, given by Peters [1]. Given an objective function

$$J = u^T N(t) u, \tag{6.15}$$

subject to the nonlinear dynamics

$$M(q)\ddot{q} = u + F(q, \dot{q}) \tag{6.16}$$

and the task constraint

$$A(q, \dot{q}, t)\ddot{q} = b(q, \dot{q}, t). \tag{6.17}$$

The solution is given by the control law

$$u = N^{-\frac{1}{2}} \left(AM^{-1}N^{-\frac{1}{2}} \right)^{+} (b - AM^{-1}F), \tag{6.18}$$

where D^{+} denotes the pseudo inverse for a general matrix D and $D^{\frac{1}{2}}$ denotes the the symmetric, positive definite matrix for which $D^{\frac{1}{2}}D^{\frac{1}{2}} = D$.

For the operational space control of this system, the task description is given by the end-effector trajectory, where

$$h(q,t) = f(q(t)) - x_d(t) = x(t) - x_d(t) = 0, \tag{6.19}$$

with $f(q(t))$ being the forward kinematics of the system that maps the joint positions $q(t)$ to the task space. Given the desired trajectory $x_d(t)$, we can use an attractor in the task space in the form

$$(\ddot{x} - \ddot{x}_d) + K_D(\dot{x} - \dot{x}_d) + K_p(x - x_d) = 0, \tag{6.20}$$

where K_D and K_P are positive-definite damping and proportional gains. Using the Jacobian relations,

$$\dot{x} = J(q)\dot{q} \tag{6.21}$$

and

$$\ddot{x} = J(q)\ddot{q} + \dot{J}(q)\dot{q}, \tag{6.22}$$

the task space constraint becomes

$$\ddot{x}_d + K_D(\dot{x} - \dot{x}_d) + K_p(x - x_d) = J(q)\ddot{q} + \dot{J}(q)\dot{q}, \tag{6.23}$$

which can be put in the form

$$A(q,\dot{q},t)\ddot{q} = b(q,\dot{q},t), \tag{6.24}$$

with

$$A(q, \dot{q}, t) = J \tag{6.25}$$

and

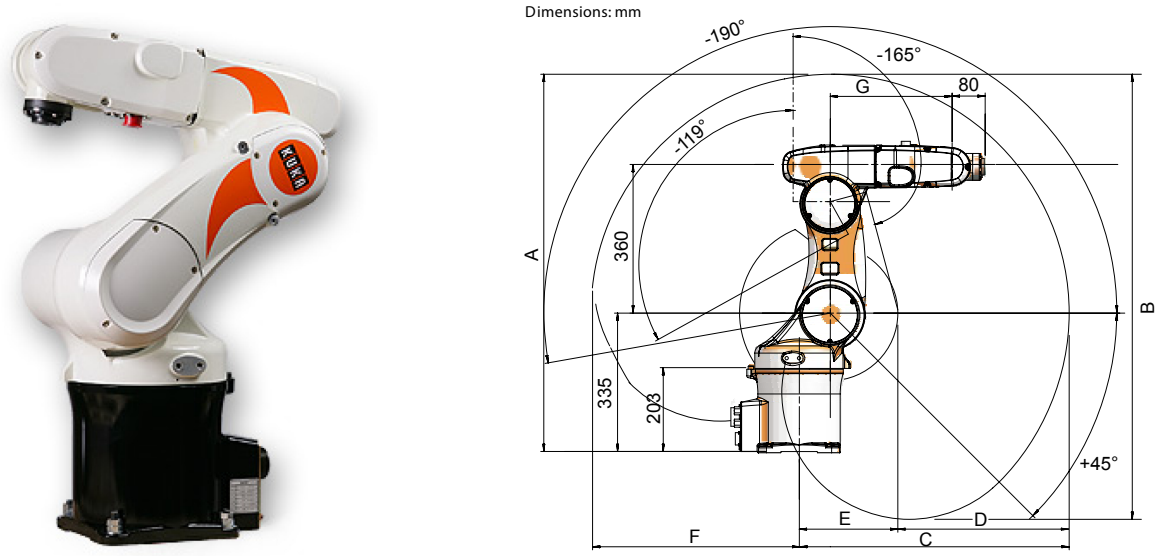
$$b(q, \dot{q}, t) = \ddot{x}_d + K_D(\dot{x} - \dot{x}_d) + K_p(x - x_d) - \dot{J}(q)\dot{q}. \tag{6.26}$$

As discussed in Peters [1], the choice of the parameter $N(t)$ in the objective function is important in influencing the behaviors of the resulting optimization. For choices $N = M^{-1}$, $N = M^{-2}$, and $N = I$, different versions of the control law (6.18) can be produced with drastically different behaviors. While the feedback control law is optimal with the assumption that the given dynamic model is perfect, actual performance of the control law differs depending on the accuracy of the given model. As discussed in another paper by Peters [136], small unmodeled nonlinearities can have a drastic effect. If the estimated mass matrix differs by a small amount, the control law will result in unstable null-space behaviors. We refers the readers to Peters [136] for a detailed discussion on robustness issues for this particular model.

In this work, we are interested in extending the robust trajectory optimization-based approaches proposed in Chapter 4 and Chapter 5 to compensate for potential modeling errors in the system matrices.

6.3.2 Results

Our experimental platform is the KUKA KR 5 sixx R650 (Fig. 6.8), a popular industrial robot. It is a small-size 6R manipulator with a spherical wrist with a total weight of 127kg, 5kg of payload, and maximum stretch of 0.65m from the base. For all experiments, we implement the operational space control for this robot's end effector to track a circular



(a) KUKA KR 5 sixx R650 industrial robot

(b) Kinematic model of the KUKA R650 robot

Figure 6.8: KUKA KR 5 sixx R650 industrial robot.

motion. The X, Y, Z components of the end effector are given by sinusoidal signals

$$\begin{aligned} X &= 0.6\sin(t) \\ Y &= 0.15\sin(t) \\ Z &= 0.6\cos(t). \end{aligned} \tag{6.27}$$

As discussed previously, the dynamics for this robot takes the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau. \tag{6.28}$$

In this operational space control scenario, we model uncertainty in the mass matrix of the system in the form of a time-varying parameter variation, as discussed in [136].

Table 6.3: KUKA KR 5 sixx R650 joint limits

Axis	Range	Speed
Axis 1 (A1)	$\pm 170°$	$375°/s$
Axis 2 (A2)	$+45°/-190°$	$300°/s$
Axis 3 (A3)	$+165°/-119°$	$375°/s$
Axis 4 (A4)	$\pm 190°$	$410°/s$
Axis 5 (A5)	$\pm 120°$	$410°/s$
Axis 6 (A6)	$\pm 358°$	$660°/s$

We implemented two trajectory-based approaches using the single model and multi-model robust trajectory optimizers proposed in Chapter 4 and Chapter 5. We generate the optimal trajectories for this operational space control task, incorporating the full dynamics of the robot as well as the limits on the joint positions and velocities. These figures are given in Table 6.3. For the multiple-model trajectory optimizer, we implement one additional model, incorporating a second-order low pass filter on the control inputs, in addition to the nominal model. The filter dynamics take the form

$$\begin{aligned} \dot{x}_f &= A_f x_f + B_f u \\ \dot{u}_f &= C_f x_f + D_f u, \end{aligned} \tag{6.29}$$

as discussed in Section 6.2. Since the robot has six degrees of freedom, each model contains 12 states, with two additional states for the filter on each control input. In total, there are 36 states for the multi-model trajectory optimizer.

Both trajectory optimization methods are implemented a receding-horizon control framework similar to the setup described in Chapter 4 and Chapter 5. We compute trajectories for the system, apply the obtained control inputs for a fixed duration, then compute new trajectories using the current states as initial conditions. For all both approaches, the total

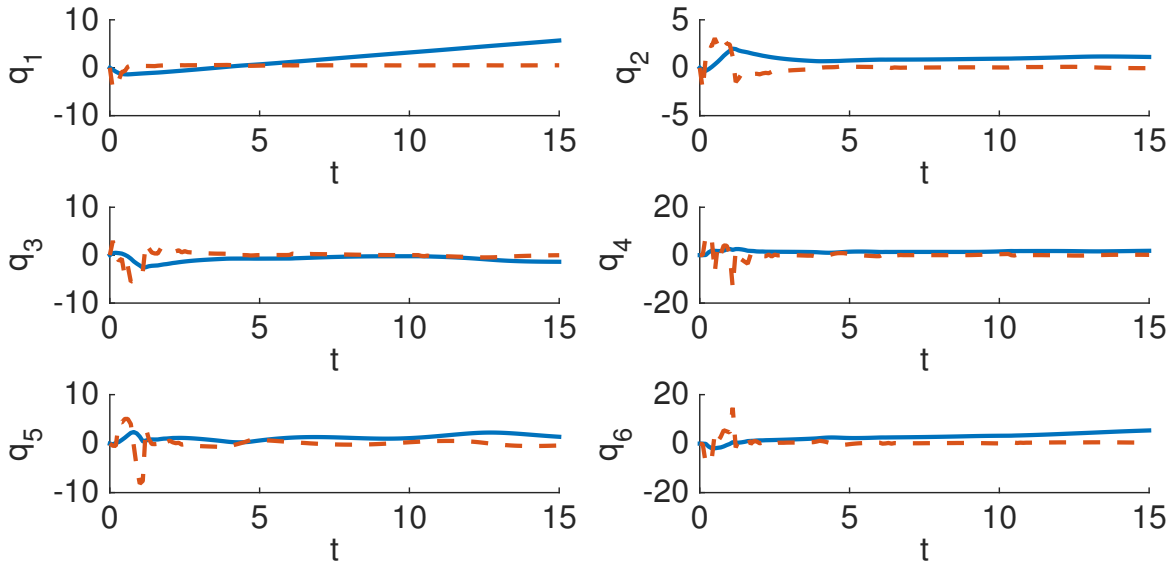


Figure 6.9: A snapshot of the optimized joint trajectories for the KUKA KR 5 sixx R650 circular task. Joint positions (blue) and velocities (red).

simulation time is 15s with a horizon of 0.1s, with 30 collocation points for each computed trajectory. Fig. 6.9 shows a snapshot of the optimized joint trajectories generated at the beginning of a horizon for one full revolution in the circular trajectory.

To compare the performance of these methods, we also implemented the optimal control-based operational space framework proposed in Peters [1]. For the objective function, we used

$$N = M^{-2}, \tag{6.30}$$

in the formulation in (6.15), which is a version of the well-known control law proposed in Khatib [137] with an additional null-space term, with the interpretation consistent with principle of virtual work of d'Alembert. We refer the readers to [1] for a detailed derivation of the control law and its interpretation.

To illustrate the effects of different types of uncertainties, three scenarios $S1$-$S3$ were

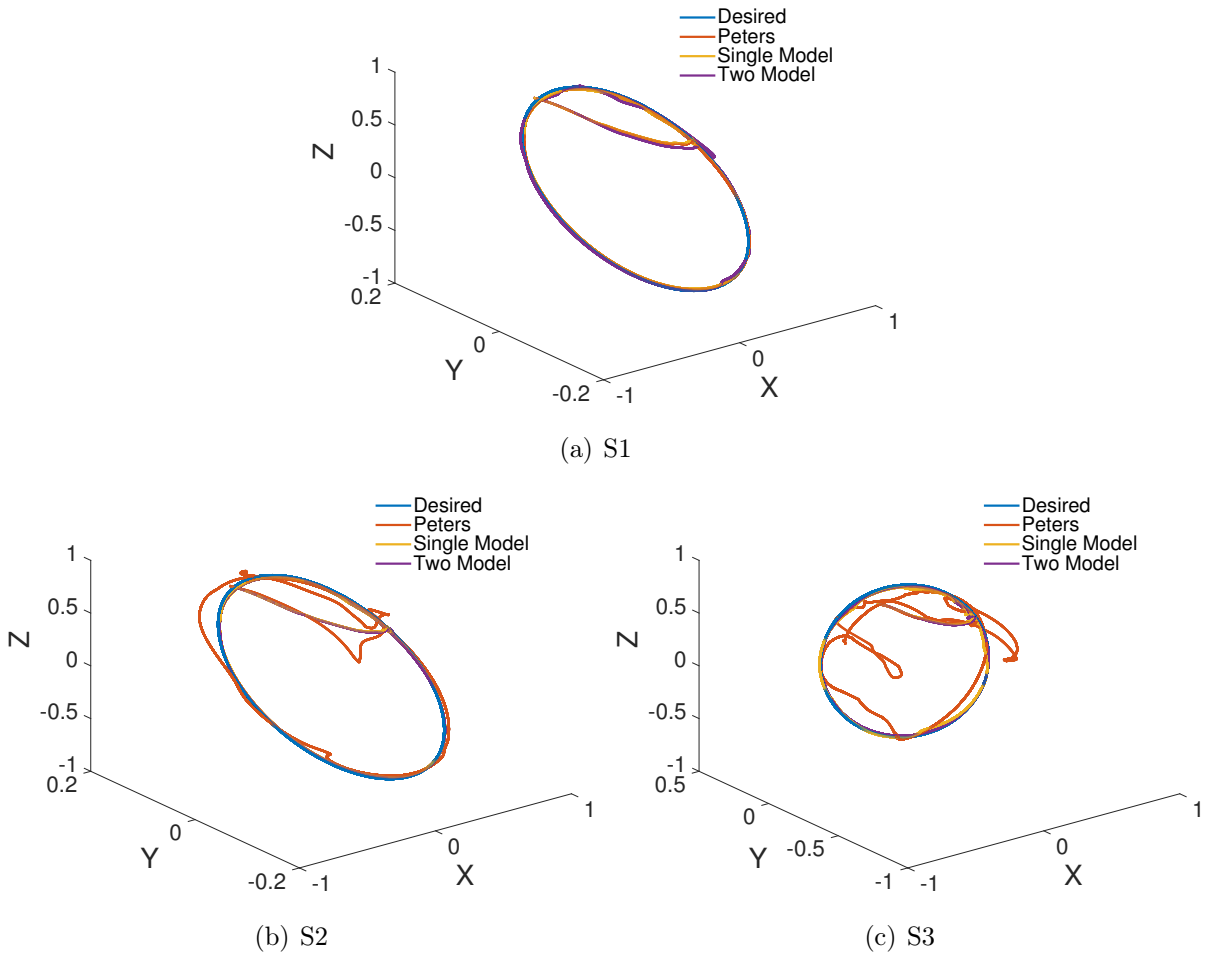
Table 6.4: Model uncertainties for the KUKA KR 5 sixx R650

	S1	S2	S3
Parametric Uncertainty	None	Sinusoidal	Sinusoidal
Unmodeled Dynamics	None	None	Time delay

set up using various combinations of initial conditions, parametric uncertainties, and unmodeled dynamics. Table 6.4 lists the conditions for the three scenarios. In S1, the three approaches were tested with no parametric uncertainties and no unmodeled dynamics. As expected, all three approaches successfully tracked the circular reference trajectories for the end effector, with comparable tracking performances. The multi-model approach deviated from the others as it has the most conservative assumptions in terms of uncertainty and performed the worst when no uncertainties were present in the system. The closed-loop response of the three approaches are shown in Fig. 6.10(a).

In S2, the three approaches were tested with a sinusoidal parametric uncertainty in the mass matrix of the manipulator (as described in Peters [136]), with no additional unmodeled dynamics. Given that this setup matches closely with the assumptions made by the single-model RHC, we expect that it will have the best performance. Indeed, optimal control law in exhibited a large tracking error while the single and multi-model achieved better tracking accuracy, with similar performances. The closed-loop response of the three approaches are shown in Fig. 6.10(b).

In S3, the three approaches were tested with the sinusoidal parametric uncertainty and a time-delay for the control inputs. Due to the significant parametric variation and the unmodeled dynamics, the optimal control law without model uncertainties exhibits significant errors in tracking the reference motion. The results show that the multi-model trajectory optimizer performed better compared to the single-model approach since it contained more expressive models of the uncertainties. The closed-loop response of the three approaches



(a) S1

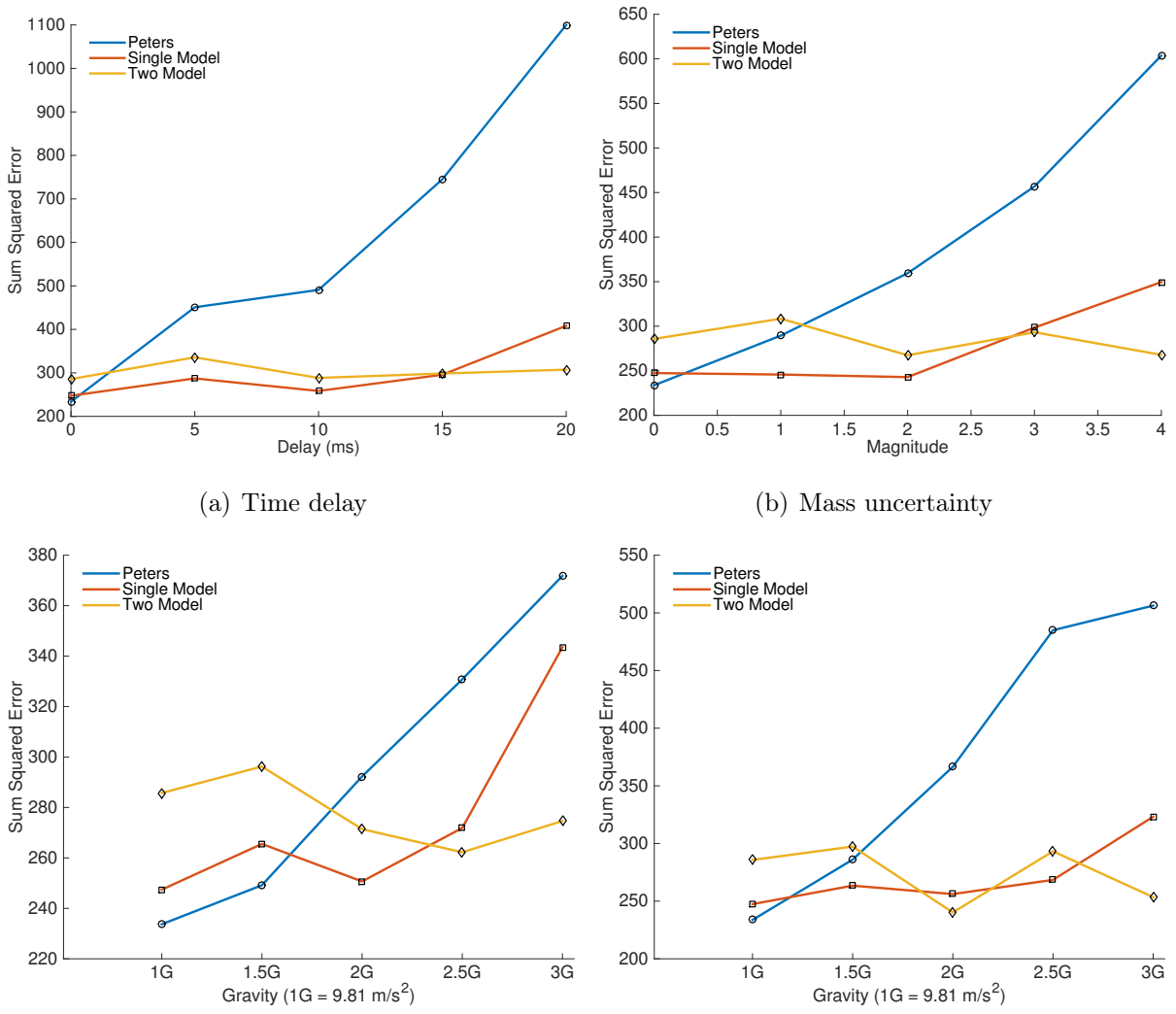
(b) S2

(c) S3

Figure 6.10: Comparison of the operational space control law of Peters [1], robust single-model RHC, and robust multi-model RHC. Four scenarios $S1$-$S3$ were setup using various combinations of initial conditions, external disturbances, parametric uncertainties, and unmodeled dynamics listed in Table 6.4

are shown in Fig. 6.10(c).

For both parametric uncertainties and unmodeled dynamics, we also conducted extensive evaluations on the performance of the proposed approaches in response to parametric



(a) Time delay

(b) Mass uncertainty

(c) Gravity

(d) Gravity

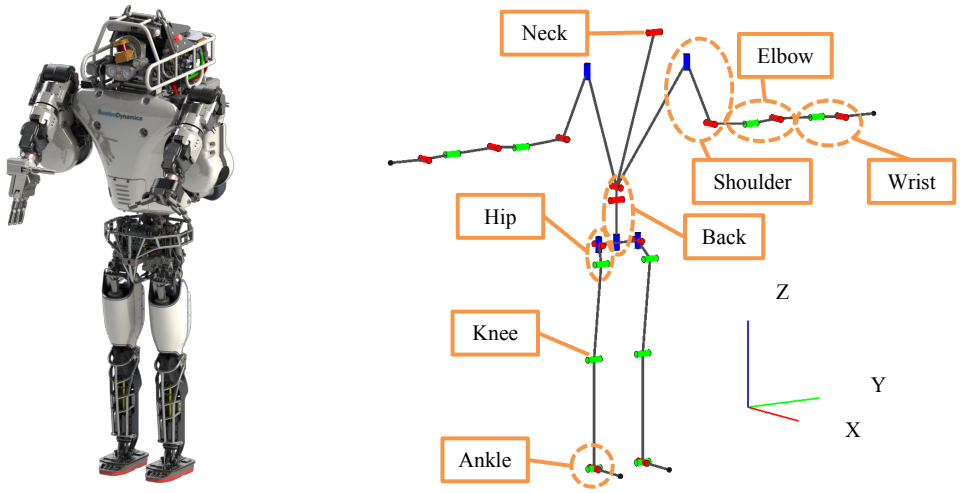
Figure 6.11: Sum of squared errors for the operational space control.

values within a specific structural class. Fig. 6.11 shows the sum of squared tracking errors for the three approaches with respect to the desired end-effector trajectory (circle). The average errors are obtained over batches of 20 simulations, and are plotted against parameter variations in a specific type of structural uncertainty, described below.

Fig. 6.11(a) shows the tracking errors with respect to the amount of delay in the system, varying from 0 to 20 ms. In the nominal model with no delay, the optimal control law outperformed both the single and two model robust RHCs. As the amount of delay is increased, the optimal control design based on the nominal model exhibited increasing tracking errors. The single and two model RHCs both performance well in the presence of delays, with a slight performance edge for the two model RHC for larger delays.

Fig. 6.11(b) shows the tracking errors with respect to a sinusoidal parametric uncertainty in the mass matrix of the manipulator. As the magnitude of the uncertainty increases, we expect robust RHCs to outperform the optimal design based on the nominal model. Indeed, the single model RHC performed the best as the designed parametric variation matched the true uncertainty in the system, while the two model RHC resulted in average performance in the spectrum, but best when the amount of uncertainty was large.

Fig. 6.11(c) and Fig. 6.11(d) show the results of another set of simulations where there is a gravity mismatch between the control design and reality. In Fig. 6.11(c) , the true gravity was held at 9.8 m/s^2 while the assumed gravity in the control design was varied from 1-3 Gs. This implies that in the gravity compensation component of the controller, more torques were supplied than the necessary amount to achieve gravity compensation. Fig. 6.11(d) simulated the opposite scenario, where the assumed gravity in the control design was held constant at 9.8 m/s^2 while the true gravity varied from 1-3 Gs. This implies that the amount of control authority in the control design was insufficient in achieving effective gravity compensating. Overall, the results are as expected - the optimal design based on the nominal model of gravity was sensitive to variations in gravity and the tracking errors correlated positively as the mismatch in gravity was increased. The single and two model RHCs performed about the same in both scenarios, effectively compensating for the variations in gravity.



(a) Atlas robot (b) Atlas kinematic model

Figure 6.12: Taken from Boston Dynamics Atlas Robot Operation and Maintenance Manual, ATLAS-01-0018-v2.0, courtesy of Boston Dynamics.

6.4 Atlas humanoid push recovery

To test the limits of the proposed robust trajectory optimizers in terms of dimensionality, we chose a reduced-order full-body model for the Atlas bipedal humanoid robot from Boston Dynamics, currently used in the DARPA Robotics Challenge. The Atlas robot is a hydraulic humanoid with near-human anthropometry. The body weighs approximately 160 kg and contains 28 hydraulically-actuated degrees of freedom, including two arms, legs, feet and a torso. The Atlas robot and its kinematic configuration is shown in Fig. 6.12. Table 6.5 contains the ranges for joint motions for each limb, in the X, Y, Z directions.

In the existing literature, studies such as Stephens [2] have shown push recovery strategies such as ankle, hip, and stepping, subject to the magnitude of the external disturbance. For small pushes, the robot can apply corrective torques at the ankle joint to compensate for the translation of the CoM. For larger disturbances, a torso rotation in conjunction with

Table 6.5: Atlas humanoid robot joint limits

	DoF	X (°)			Y (°)			Z (°)		
		Min	Max	Range	Min	Max	Range	Min	Max	Range
Neck	1	-	-	-	-34.5	+65.5	100	-	-	-
Back	3	-30	+30	60	-12.6	+30.9	43.5	-38	+38	76
Shoulder	3	-90	+90	180	-	-	-	-45	+90	135
Elbow	1	-135	0	135	0	+180	180	-	-	-
Wrist	2	-67.5	67.5	135	0	+180	180	-	-	-
Hip	3	-30	+30	60	-92.4	+37.7	130.1	-45	+10	55
Knee	1	-	-	-	0	+135	135	-	-	-
Ankle	2	-25	+25	50	-52	+28	80	-	-	-

ankle torques can significantly improve the performance of the push recovery controller. For significant disturbances that are not recoverable using the ankle and/or hip strategies, a stepping motion leads to an instantaneous shift in the CoM position and can stabilize the robot. These strategies are illustrated in Fig. 6.13.

For this reduced-order Atlas model, we are interested in obtaining human-like emergent behavior from an optimization process. Instead of manually defining recovery strategies under the ankle/hip/stepping formulation, we formulate an optimization problem that explicitly models a bounded external disturbance and other internal parametric uncertainties. We show that under this optimization-based approach, we can generate recovery behaviors automatically, depending on the assumed upper bound of the disturbance.

The mathematical modeling of the dynamics is similar to the three-link and six-link models described in the previous sections. We also model contacts on the robot, which requires additional constraints on the system. We follow the approach used in Posa [138]. The joint space of the robot is defined in generalized coordinates, which in this case, is

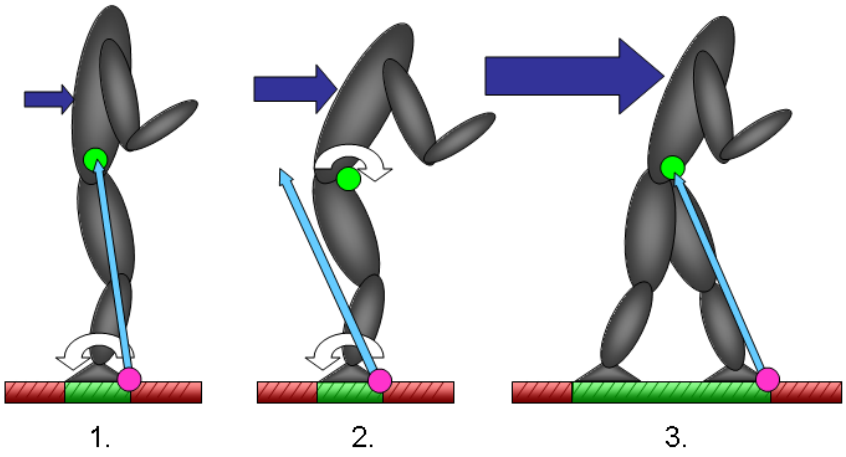


Figure 6.13: Ankle, hip, and stepping strategies for push recovery. Illustration courtesy of Ben Stephens [2].

individual actuated degree of freedom in the robot. The nonlinear equations of motion is similar to (6.3), in the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = B(q)\tau + J^T\lambda, \tag{6.31}$$

where M is the *mass matrix* that is square, symmetric, and positive definite; C consists of centripetal and Coriolis forces; G represents the gravitational and other forces acting on the robot; B is the input mapping for τ, a vector of torques that are the controlled inputs to the system; λ is a vector of constraint forces acting along the surface normal, which is projected by the Jacobian matrix J into the joint space as

$$J(q) = \frac{\partial\phi(q)}{\partial q}, \tag{6.32}$$

As with (6.3), this equation can be placed in a form consistent with

$$\begin{aligned} \dot{x}(t) &= [f(x) + \Delta f(x, \theta, t)] + g_1(x)w(t) + [g_2(x) + \Delta g_2(x, \theta, t)]\, u(t) \\ z(t) &= [h_1(x), u(t)]^T \\ x(t_0) &= x_0, \end{aligned} \tag{6.33}$$

subject to the non-penetration constraint

$$\phi(q) \geq 0, \tag{6.34}$$

where the equality

$$\phi_i(q) = 0 \tag{6.35}$$

hold iff the i-th constraint is active. The complementarity constraint

$$\begin{aligned} \lambda &\geq 0 \\ \phi^T(q)\lambda &= 0 \end{aligned} \tag{6.36}$$

ensures that the contact forces can be non-zero iff the bodies are in contact.

For the purposes of dimensionality reduction in generating locomotive behaviors, we lock down the upper body of the robot and treat the neck, shoulder, elbow and wrist joints of the robot as fixed. The remaining 15 degrees of freedom consists of the back (3), hip (6), knee (2), and ankle (4). The state vector of the system, which consists of q and $\dot{q}$, contains 30 states. The actual equations of motion are generated using the Dynamic Animation and Robotics Toolkit (DART), which computes Lagrange's equations derived from D'Alembert's principle, with full access to internal kinematic and dynamic quantities, such as the mass matrix, Coriolis and centrifugal forces.

To solve this 30-dimensional optimization problem, we first follow the procedure de-

tailed in Posa [138] and Anitescu [139], which relaxes constraints of the form

$$\begin{aligned} G(x) &\geq 0 \\ H(x) &\geq 0 \\ G^T(x)H(x)\lambda &= 0 \end{aligned} \tag{6.37}$$

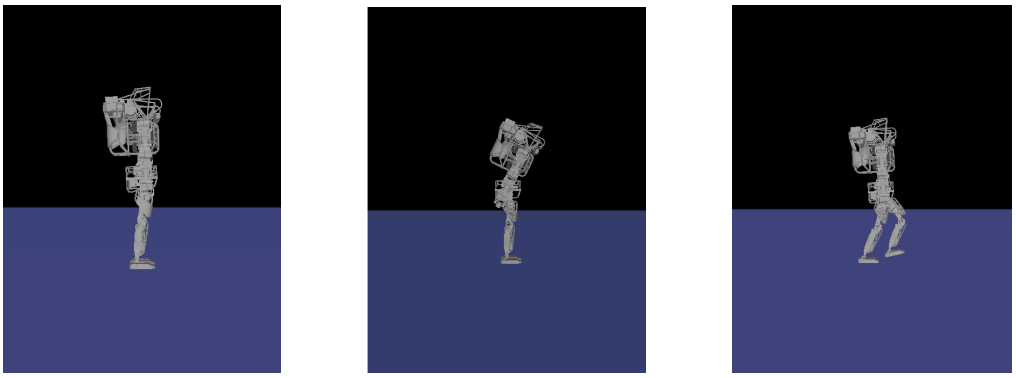
into the form

$$\begin{aligned} G(x) &\geq 0 \\ H(x) &\geq 0 \\ G_i(x)H_i(x)\lambda &\leq 0, \end{aligned} \tag{6.38}$$

where a point-wise evaluation is used. In addition, to improve numerical stability, we introduce additional slack variables α and β in the form

$$\begin{aligned} \alpha, \beta &\geq 0 \\ \alpha &= G(x) \\ \beta &= H(x) \\ \alpha_i\beta_i &\leq 0. \end{aligned} \tag{6.39}$$

Overall, this optimization framework, as applied to the reduced-order model of Atlas, results in the generation of different push recovery strategies according to the assumed bounds of the external disturbances and other internal parametric variations. Fig. 6.14 shows several examples of generated strategies. Unlike manually defined push recovery control, these motions are generated using the whole-body model of the robot without explicit reference to any particular joint. As a result, multiple joints are often used simultaneously to achieve an overall behavior, and we use the dominant joint motions to broadly

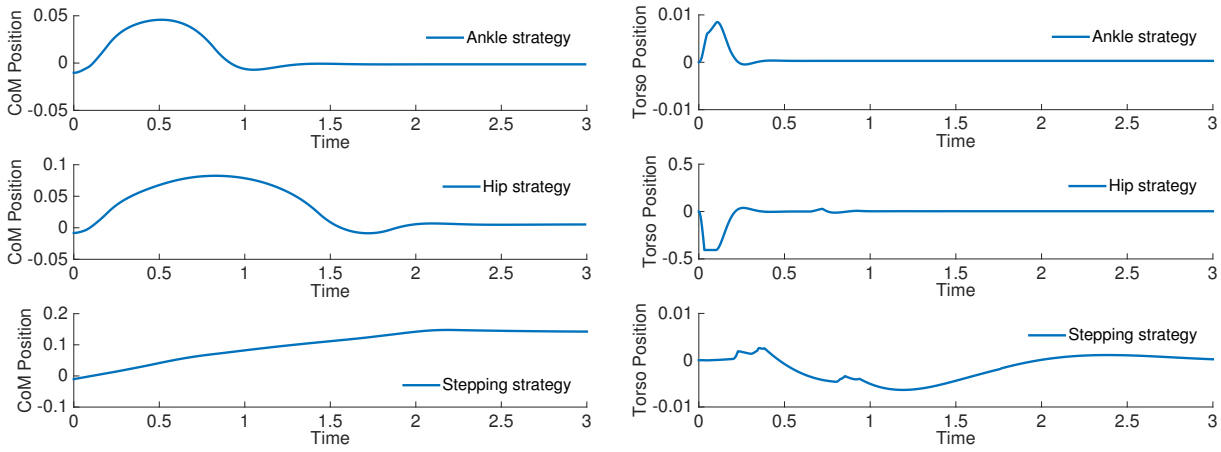


(a) Ankle strategy (b) Hip strategy (c) Stepping strategy

Figure 6.14: Snapshot of optimized behaviors for push recovery. The ankle strategy compensates for small external disturbances by using ankle torques. The hip strategy rotates the torso in the direction of the push to recover from medium disturbances. For large disturbances, stepping forwards shifts the CoM location forward into the support polygon.

interpreted behaviors as ankle, hip, and stepping strategies in the description below.

Fig. 6.15(a) shows the the center of mass trajectories in the forward direction in each of the three generated strategies, corresponding to the screenshots in Fig. 6.14. For small external disturbances, the "ankle" strategy accommodates forward CoM translations of around 0.05m, using mainly torques applied at the ankle. For moderate external disturbances which results from a forward CoM motion of 0.08m, the "hip" strategy is more effective in stabilizing the body back into equilibrium. For large disturbances, the robot achieves stabilization by stepping, and instead of regulating the CoM back to the original, the CoM moves forward to the 0.14m position. Fig. 6.15(b) shows the corresponding torso motions for each of the strategies. In both the ankle and stepping strategies, there is little to no motion for the torso in the forward direction. In the hip strategy, the robot actively rotates the torso forward by about 0.4 rad, then regulates the torso position back to the equilibrium states.



(a) Center-of-Mass trajectories

(b) Forward toroso (mtorso) trajectories

Figure 6.15: Center of mass and torso trajectories for the ankle, hip, and stepping strategies. In both the ankle and stepping strategies, there is little to no motion for the torso in the forward direction. In the hip strategy, the robot actively rotates the torso forward by about 0.4 rad, then regulates the torso position back to the equilibrium states.

6.5 Discussion

In all of the example problems demonstrated in this chapter, trajectory optimization within a receding horizon control framework was shown to be an effective approach in robustifying controller design against external disturbances, parametric uncertainties, and unmodeled dynamics. In particular, the use of multiple models is an effective and flexible way of representing uncertainties, where each model can have entirely different structural properties. For example, In the multi-model RHC case in Fig. 6.6(d), the nominal model was augmented by a second, higher dimensional model, which accounted for high-order dynamics. The concatenation of a different state space would not have been possible with a single model.

Since the proposed approaches are based on approximate methods using the spectral

approximation framework presented in Chapter 2, the solutions obtained only approximates the true globally optimal solutions that can be obtained by solving the robust control problem analytically. Hence, it is important to characterize the quality of the solutions obtained. The quality of the solution is controlled by two factors: the initial guesses supplied to the solver, and the number of collocation points used in approximating the trajectories.

The single and multi-model RHC are both based on direct trajectory optimization presented in Section 4.3, which place a softer requirement on the quality of the initial guesses as compared to the indirect methods presented in Section 4.2. For low-dimensional problem such as Section 6.2 and Section 6.3, we supplied random initial guesses to the solver to produce the trajectories. Due to the hybrid nature of the push recovery experiments in Section 6.4, the requirements for initial guesses, especially in the stepping strategy, was more important. In general, for larger problems, the quality of the initial guess will be important, and as discussed in [140], the obtained trajectories tend to be homotopic to the initial trajectories and it is difficult to recover from an initial guess in a bad homotopy class.

The second factor in the solution quality is the number of collocation points used in approximating the trajectories. In Fig. 6.16, we examined the metric for solution quality, the objective function value, with respect to the number of collocation points used. For the problem setups in Section 6.2, with a state-space dimension of up to 18, the optimal number of collocation points is 20, beyond which no improvements to the objective function value can be obtained. For larger scale problems, especially those that are highly nonlinear, we expect that a higher number of collocation points will be required, though it is difficult to provide a quantitative analysis as the results are highly problem-specific.

The number of collocation points is an important factor in the computational efficiency of the trajectory optimizers. Computational efficiency represents a trade-off with solution

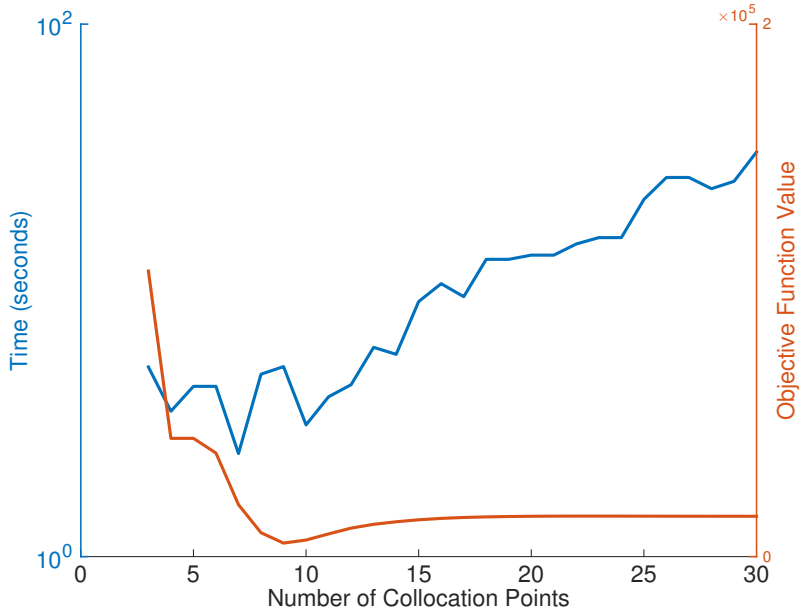


Figure 6.16: Computation time (seconds) and objective function value vs. the number of collocation points.

quality, since as the number of collocation points is increased, computational cost also increases. In Fig. 6.16, we examined the computation time with respect to the number of collocation points used and obtained a super-linear relationship. The increase from 3 to 30 collocation points led to a corresponding linear increase in the computation time, from 5 to 33 seconds. We expect this relationship to hold for larger-scale problems.

For low-dimensional problem presented in Section 6.2 and Section 6.3, no initial guesses were required to generate the solutions in any of the batch simulations conducted. This is a clear advantage of a direct trajectory optimization setup, as opposed to the indirect variety using first-order necessary conditions (Section 4.2). In Section 6.4, the 30-dimensional optimization problem required initial guesses in order the generate the three strategies in Fig. 6.13. This is likely due to the complexities of the underlying dynamics, in terms of both nonlinearity and dimensionality. For large systems such as this, we expect that good initial guesses will be an important component, even for direct trajectory optimization.

There is a clear need for both effective initial guesses and reducing the computational burdens associated with a large number of collocation points. An effective strategy is to start out the optimization with random initial guesses, with a low number of collocation points, then using the optimization result to jump start a second optimization iteration with a high number of collocation points. In high dimensional examples such as those in Section 6.4, we found this to be a particularly effective way of producing high quality solutions while reducing computation time.

An important drawback to the proposed trajectory optimization-based RHC is the computational cost. All experiments in Section 6 were conducted in MATLAB with a Quad Core desktop computer with 8GB of RAM. A single trajectory for a six-dimensional problem in Section 6.2 requires 45s to compute, the 12-dimensional problem in Section 6.3 requires 25m, and the 30-dimensional problem in Section 6.4 require 7h. These figures are dependent on the number of collocation points used, and the computational efficiency of the multiple-model versions of these problems scale near linearly with respect to dimensionality. These numbers are consistent with the computation times reported in other trajectory-optimization based approaches [141], but clearly, they cannot currently be implemented in a real-time RHC setting. The target implementation for these trajectory-based approaches is atrajectory library approach such as [142], where the expensive trajectory optimization step is computed offline and interpolated online.

There are several reasons why the proposed robust trajectory optimization implementations are slow. First, the underlying optimization problem is difficult in that it is an adversarial optimization problem, which is more harder to compute compared to a traditional NLP-based trajectory optimizer. In Chapter 4, we have presented techniques to reduce a nested optimization setup, which already improved the computational efficiency, but there remains additional computational burden due to the nature of the problem formulation. Second, the examples presented in this chapter have varying levels of dynamics - from simple dynamics in Section 6.2 that can be derived by hand, to complicated

multi-body systems in Section 6.4 that must be computed algorithmically. The underlying computational times factored in the computation for the dynamics and the actual computational complexity of the trajectory optimizer will likely be reduced given simple dynamics. Finally, the software implementation based on MATLAB very likely impacted the computational efficiency significantly. Given a native implementation and multi-core setup, the underlying RHC can be parallelized effectively to obtain a linear speed-up.

Despite the seemingly inefficient computation, the proposed trajectory optimization remains orders of magnitude faster than previous implementations. A alternative implementation of trajectory optimization without using the complementarity conditions in (4.25) is solved in two parts according to (6.7): an *inner* maximization problem and an *outer* minimization problem. The running time for this naive approach is approximately 12 hours for a six-dimensional problem, compared to under one minute for our MCP formulation, for the same number of collocation points.

A key assumption in the differential game formulation to robust control is the notation of *rationality* - that this adversarial optimization problem is played to the equilibrium state by rational decision-makers u^* and w^*. In reality, not all robust control situations satisfy this assumption, which ultimately lead to sub-optimal controller performance, or controller *conservatism*.

First, the stabilizing player assumes the existence of an opposing player. When this assumption is violated, for example, in the case of no uncertainties or modeling errors in the system, then the controller performance is sub-optimal. This can be seen in Fig. 6.11, where conservative robust control designs based on single or model models significantly under performs an optimal control design. Theoretically, this is well understood in that the goal of any robust control design is not to perform best given any one model, but to perform well in the average case, across a spectrum of modeling errors. In the proposed framework, we can guarantee that without an opposing player, the minimization control

design will be stable, which directly arises from the definition of the $\mathcal{L}_2$-gain in (3.27). We cannot guarantee the level of conservatism in the control design, which is a function of the assumed magnitude of the system uncertainties.

Similarly, in the case of an irrational opponent, we can only guarantee stability, but not the level of conservatism in the control design. In the existing literature for decision theory, the issue of irrationality has been addressed extensively. Unlike the classical models of expected utility [143][144], observed decision-making behaviors often exhibit systematic deviations, such as risk/loss aversio, inaccurately weighting low/high probability events, and displaying greater disutility for losses compared to equal-amount gains. Prospect theory [145] is a descriptive model that incorporates many irrational biases and can serve as a starting point for stochastic models. In this work, with the dynamics being deterministic as opposed to stochastic, it is unclear what alternative formulations for irrationality can be. One potential idea is the formulation of stochastic dynamics for robust control, where two opposing players can elect to have irrational strategies through the stochastic component portion of the system.

6.6 Summary

In this chapter, we provided experimental validations of the robust control designs proposed in Chapter 3, Chapter 4, and Chapter 5. Through three example related to the control of humanoid robots, we tested the proposed control designs under varying degrees of complexity for the underlying dynamic systems. The results show that the proposed robust trajectory optimizers are effective in compensating for uncertainties such as external disturbances, parametric variations, and unmodeled dynamics, even for complex nonlinear systems with high dimensionality. We provided an extensive discussion of the computational aspects of robust trajectory optimization and highlighted the significant

achievements in tractable designs for robust control.

7 Conclusions

7.1 Overview

In this thesis, we explored the topic of augmenting feedback control designs to compensate for uncertainties, both internal and external to a dynamic system. This problem was approached in a new direction: instead of handling uncertainties *implicitly* or learning from data, we can formulate *explicit* mathematical models of uncertainties and achieve robustness in control design. Specifically, this thesis addressed three keys questions:

- Modeling: how do we model uncertainties such as external disturbances, parametric variations, and unmodeled dynamics in a nonlinear dynamic system?
- Optimization: how do we extend a traditional nonlinear optimal control framework to consider these uncertainties?
- Computation: How do we solve this new robust optimization problem in a computationally tractable way for nonlinear systems with high dimensionality?

7.2 Design guidelines

This thesis, along with many other recent advances in robust control, have opened the possibility of designing robust controllers for practical applications. For potential practitioners

of robust control, we have the following design guidelines on the appropriate methods to consider given specific parameters for the underlying system. We focus our discussion on four aspects of system dynamics: the nonlinearity of the dynamics, the complexity of the equations of motion, the dimensionality of the state space, and the inclusion of constraints for the states and inputs.

Table 7.1: Robust control design guidelines

Dynamics	Complexity	Dimensionality	Constraints	Design
Linear	-	Low	None	SOS or RHC
Linear	-	High	None	RHC
Linear	-	-	Yes	Treat as nonlinear system
Nonlinear	Any	Low	Any	Dynamic programming
Nonlinear	Simple	High	None	Indirect trajectory optimization
Nonlinear	Simple	High	Yes	Direct trajectory optimization
Nonlinear	Complex	High	No	Direct trajectory optimization
Nonlinear	Complex	High	Yes	Direct trajectory optimization

For linear systems or nonlinear systems that can be linearized, existing tools such as linear-matrix inequalities and sum-of-squares optimization can be used to formulate an approximate feedback control design based on analytical closed-formed expressions. These techniques are reviewed in Section 3.3. For high dimensional systems, receding-horizon control is an effective way of combining simple controller to achieve robustness (see Section 1.2.6). For linear systems with constraints, in most cases, it should be treated as a nonlinear system.

For nonlinear systems where the dimensionality of the state space is low, a dynamic programming-based approach can be considered, regardless of other factors. While the global necessary and sufficient conditions derived in Section 3.2 would be difficult to solve,

even for simple problems, numerical approaches based on either direct policy/value iteration or a value function approximation scheme (see Section 3.3) can be used if the dimensionality of the problem is sufficiently small.

For high-dimensional nonlinear systems, a trajectory optimization approach is recommended. When the dynamics is simple and without associated state/input constraints, an indirect approach (see Section 4.2) can be used. For all other cases, direct trajectory optimization (see Section 4.3) is the only available tool for control design.

In terms of uncertainties, the dynamic programming and trajectory optimization-based approaches can address structural uncertainties, both internal and external to the system. For unmodeled dynamics, a multi-model design is appropriate (see Chapter 5).

7.3 Future work

We believe that continued work along these lines will result in more effective robust control designs for nonlinear systems to enable greater autonomy for complex robots such as humanoids. This work is a general framework which can be extended in many ways, incorporating other ways of modeling norm-bounded uncertainties and alternative formulations of differential games with different underlying assumptions of available information. The proposed trajectory optimization method can be used in conjunction with a global planner, which can be extended to incorporate environmental constraints and uncertainties. Outside robustness issues, the proposed approaches can be extended to other forms of differential games, such as pursuit evasion games, and demonstrate applicability in domains such as economics and finance.

References

[1] J. Peters, M. Mistry, F. Udwadia, R. Cory, J. Nakanishi, and S. Schaal, "A Unifying Methodology for the Control of Robotic Systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug. 2005, pp. 1824–1831.

[2] B. Stephens, "Humanoid Push Recovery," in *IEEE-RAS International Conference on Humanoid Robots*, Nov. 2007, pp. 589–595.

[3] M. Xie, *Fundamentals of Robotics: Linking Perception to Action*, 1st ed., ser. Machine Perception and Artifical Intelligence. World Scientific Publishing Company, 2003.

[4] K. Yamane and J. Hodgins, "Simultaneous Tracking and Balancing of Humanoid Robots for Imitating Human Motion Capture Data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 2510–2517.

[5] J. C. Doyle, "Guaranteed Margins for LQG Regulators," *IEEE Transactions on Automatic Control*, vol. 23, no. 4, pp. 756–757, Aug. 1978.

[6] G. Zames, "Feedback and Optimal Sensitivity: Model Reference Transformations, Multiplicative Seminorms, and Approximate Inverses," *IEEE Transactions on Automatic Control*, vol. 26, no. 2, pp. 301–320, Apr. 1981.

[7] B. A. Francis and G. Zames, "On H-Infinity Optimal Sensitivity Theory for SISO Feedback Systems," *IEEE Transactions on Automatic Control*, vol. 29, no. 1, pp. 9–16, Jan. 1984.

[8] S. P. Bhattacharyya, H. Chapellat, and L. H. Keel, *Robust Control : the Parametric Approach*, 1st ed. Prentice Hall, 1995.

[9] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*, 1st ed. Prentice Hall, 1996.

[10] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control : Analysis and Design*, 2nd ed. Wiley, 2005.

[11] G. E. Dullerud and F. G. Paganini, *A Course in Robust Control Theory : a Convex Approach*, 1st ed. Springer, 2000.

[12] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2001.

[13] R. Bellman, "Dynamic Programming and Lagrange Multipliers," *Proceedings of the National Academy of Sciences*, vol. 42, no. 10, p. 767, 1956.

[14] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: Optimization, Estimation and Control*, 1st ed. Taylor & Francis, 1975.

[15] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, 1st ed., ser. Modern Birkhäuser Classics. Birkhäuser, 2008.

[16] R. Isaacs, *Differential Games: a Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*, 1st ed. Courier Dover Publications, 1965.

[17] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, 1st ed. Society for Industrial and Applied Mathematics, 1995, vol. 200.

[18] T. Başar and P. Bernhard, *H-Infinity Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach*, 2nd ed. Birkhäuser, 2008.

[19] J. Engwerda, *LQ Dynamic Optimization and Differential Games*, 1st ed. Wiley, 2005.

[20] T. L. Friesz, *Dynamic Optimization and Differential Games*, 1st ed., ser. International Series in Operations Research & Management Science. Springer, 2010, vol. 135.

[21] D. Georges, "Solutions of Nonlinear Optimal Regulator and H-Infinity Control Problems via Galerkin Methods," *European Journal of Control*, vol. 2, no. 3, pp. 211–226, 1996.

[22] R. W. Beard, G. N. Saridis, and J. T. Wen, "Galerkin Approximations of the Generalized Hamilton-Jacobi-Bellman Equation," *Automatica*, vol. 33, no. 12, pp. 2159–2177, 1997.

[23] R. W. Beard, "Successive Galerkin Approximation Algorithms for Nonlinear Optimal and Robust Control," *International Journal of Control*, vol. 71, no. 5, pp. 717–743, 1998.

[24] R. W. Beard, G. N. Saridis, and J. T. Wen, "Approximate Solutions to the Time-Invariant Hamilton–Jacobi–Bellman Equation," *Journal of Optimization Theory and Applications*, vol. 96, no. 3, pp. 589–626, 1998.

[25] M. Alamir, "Solutions of Nonlinear Optimal and Robust Control Problems via a Mixed Collocation/DAE's Based Algorithm," *Automatica*, vol. 37, no. 7, pp. 1109–1115, 2001.

[26] H. C. Ferreira, P. H. Rocha, and R. M. Sales, "Nonlinear H-Infinity Control and the Hamilton-Jacobi-Isaacs Equation," in *IFAC World Congress*, 2008, p. 188.

[27] P. Tsiotras, M. Corless, and M. A. Rotea, "An L2 Disturbance Attenuation Solution to the Nonlinear Benchmark Problem," *International Journal of Robust and Nonlinear Control*, vol. 8, no. 4-5, pp. 311–330, 1998.

[28] M. J. Grimble, "Factorised H-Infinity Control of Nonlinear Systems," *International Journal of Control*, vol. 85, no. 7, pp. 964–982, 2012.

[29] Y. Feng, M. Rotkowitz, and B. D. Anderson, "An Iterative Procedure to Solve HJBI Equations in Nonlinear H-Infinity Control," in *IFAC World Congress*, 2008.

[30] Y. Feng, B. D. Anderson, and M. Rotkowitz, "A Game Theoretic Algorithm to Compute Local Stabilizing Solutions to HJBI Equations in Nonlinear Control," *Automatica*, vol. 45, no. 4, pp. 881–888, Apr. 2009.

[31] M. J. Kim, Y. Choi, and W. K. Chung, "Bringing Nonlinear H-Infinity Optimality to Robot Controllers," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 682–698, June 2015.

[32] J. W. Helton and M. R. James, *Extending H-infinity Control to Nonlinear Systems*, ser. Control of Nonlinear Systems to Achieve Performance Objectives. SIAM, Jan. 1999.

[33] M. D. S. Aliyu, *Nonlinear H-Infinity Control, Hamiltonian Systems and Hamilton-Jacobi Equations*, 1st ed. CRC Press, 2011.

[34] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific, 2007.

[35] O. Sigaud and O. Buffet, *Markov Decision Processes in Artificial Intelligence: MDPs, Beyond MDPs and Applications*, 1st ed. Wiley, 2010.

[36] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement Learning in Robotics: a Survey," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[37] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed., ser. Wiley Series in Probability and Statistics. Wiley, 2011.

[38] I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska, "A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 6, pp. 1291–1307, Nov. 2012.

[39] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*, 1st ed. CRC press, 2010.

[40] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks Are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[41] B. Hammer and K. Gersmann, "A Note on the Universal Approximation Capability of Support Vector Machines," *Neural Processing Letters*, vol. 17, no. 1, pp. 43–53, 2003.

[42] W. T. Miller, R. S. Sutton, and P. J. Werbos, *Neural Networks for Control*, 1st ed. MIT Press, 1990.

[43] H. Zhang, D. Liu, Y. Luo, and D. Wang, *Adaptive Dynamic Programming for Control: Algorithms and Stability*, 1st ed. Springer, 2013.

[44] A. Al-Tamimi, M. Abu-Khalaf, and F. L. Lewis, "Adaptive Critic Designs for Discrete-Time Zero-Sum Games with Application to H-Infinity Control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 37, no. 1, pp. 240–247, Feb. 2007.

[45] H. Li, D. Liu, and D. Wang, "Integral Reinforcement Learning for Linear Continuous-Time Zero-Sum Games with Completely Unknown Dynamics," *IEEE Transactions on Automation Science and Engineering*, vol. PP, no. 99, pp. 1–9, 2014.

[46] D. Liu, H. Li, and D. Wang, "Neural-Network-Based Zero-Sum Game for Discrete-Time Nonlinear Systems via Iterative Adaptive Dynamic Programming Algorithm," *Neurocomputing*, vol. 110, pp. 92–100, 2013.

[47] H. Zhang, Q. Wei, and D. Liu, "An Iterative Adaptive Dynamic Programming Method for Solving a Class of Nonlinear Zero-Sum Differential Games," *Automatica*, vol. 47, no. 1, pp. 207–214, 2011.

[48] T. Dierks and S. Jagannathan, "Optimal Control of Affine Nonlinear Continuous-Time Systems Using an Online Hamilton-Jacobi-Isaacs Formulation," in *IEEE Conference on Decision and Control*, Dec. 2010, pp. 3048–3053.

[49] H. Zhang, L. Cui, and Y. Luo, "Near-Optimal Control for Nonzero-Sum Differential Games of Continuous-Time Nonlinear Systems Using Single-Network ADP," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 206–216, Feb. 2013.

[50] M. Abu-Khalaf, F. L. Lewis, and J. Huang, "Neurodynamic Programming and Zero-Sum Games for Constrained Control Systems," *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1243–1252, July 2008.

[51] ——, "Policy Iterations on the Hamilton-Jacobi-Isaacs Equation for H-Infinity State Feedback Control With Input Saturation," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1989–1995, Dec. 2006.

[52] K. G. Vamvoudakis and F. L. Lewis, "Online Solution of Nonlinear Two-Player Zero-Sum Games Using Synchronous Policy Iteration," *International Journal of Robust and Nonlinear Control*, vol. 22, no. 13, pp. 1460–1483, 2012.

[53] S. Mehraeen, T. Dierks, S. Jagannathan, and M. L. Crow, "Zero-Sum Two-Player Game Theoretic Formulation of Affine Nonlinear Discrete-Time Systems Using Neural Networks," in *International Joint Conference on Neural Networks*, July 2010, pp. 1–8.

[54] H. Zhang, L. Cui, X. Zhang, and Y. Luo, "Data-Driven Robust Approximate Optimal Tracking Control for Unknown General Nonlinear Systems Using Adaptive Dynamic Programming Method," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 2226–2236, Dec. 2011.

[55] H.-N. Wu and B. Luo, "Neural Network Based Online Simultaneous Policy Update Algorithm for Solving the HJI Equation in Nonlinear H-Infinity Control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 12, pp. 1884–1895, Dec. 2012.

[56] B. Luo, H.-N. Wu, and T. Huang, "Off-Policy Reinforcement Learning for H-Infinity Control Design," *IEEE Transactions on Cybernetics*, vol. 45, no. 1, pp. 65–76, Jan. 2015.

[57] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[58] S. Levine, "Exploring Deep and Recurrent Architectures for Optimal Control," in *Advances in Neural Information Processing Systems, Deep Learning Workshop*, 2013.

[59] Y. Wang, B. O'Donoghue, and S. Boyd, "Approximate Dynamic Programming via Iterated Bellman Inequalities," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 10, pp. 1472–1496, 2014.

[60] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed. Cambridge University Press, 2004.

[61] P. A. Parrilo, "Semidefinite Programming Relaxations for Semialgebraic Problems," *Mathematical Programming*, vol. 96, no. 2, pp. 293–320, 2003.

[62] L. Vandenberghe and S. Boyd, "Semidefinite Programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[63] S. Boyd, L. el Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, 1st ed., ser. Studies in Applied and Numerical Mathematics. Society for Industrial and Applied Mathematics, 1997.

[64] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*, 1st ed., ser. Studies in Applied and Numerical Mathematics. Society for Industrial and Applied Mathematics, July 1995.

[65] G. Chesi, *Domain of Attraction: Analysis and Control via SOS Programming*, 1st ed., ser. Lecture Notes in Control and Information Sciences. Springer, 2011.

[66] I. R. Petersen and R. Tempo, "Robust Control of Uncertain Systems: Classical Results and Recent Developments," *Automatica*, vol. 50, no. 5, pp. 1315–1335, 2014.

[67] H. Ichihara, "State Feedback Synthesis for Polynomial Systems with Bounded Disturbances," in *IEEE Conference on Decision and Control*, Dec. 2008, pp. 2520–2525.

[68] H.-J. Ma and G.-H. Yang, "Fault-Tolerant Control Synthesis for a Class of Nonlinear Systems: Sum of Squares Optimization Approach," *International Journal of Robust and Nonlinear Control*, vol. 19, no. 5, pp. 591–610, 2009.

[69] P. Gahinet and P. Apkarian, "A Linear Matrix Inequality Approach to H-Infinity Control," *International Journal of Robust and Nonlinear Control*, vol. 4, no. 4, pp. 421–448, 1994.

[70] A. Poznyak, A. Polyakov, and V. Azhmyakov, *Attractive Ellipsoids in Robust Control*. Cham: Springer, Sept. 2014.

[71] W.-M. Lu and J. C. Doyle, "H-Infinity Control of Nonlinear Systems: a Convex Characterization," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1668–1675, Sept. 1995.

[72] E. Prempain, "An Application of the Sum of Squares Decomposition to the L2 Gain Computation for a Class of Nonlinear Systems," in *IEEE Conference on Decision and Control*, Dec. 2005, pp. 6865–6868.

[73] J.-L. Wu, "Robust H-Infinity Control for Polytopic Nonlinear Control Systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2957–2962, Nov. 2013.

[74] X. Wei, L. Liu, and X. Liu, "Nonlinear H-Infinity Control Using Sum of Squares Decomposition," in *IEEE International Conference on Control and Automation*, May 2007, pp. 233–238.

[75] J. Löfberg, "Automatic Robust Convex Programming," *Optimization Methods and Software*, vol. 27, no. 1, pp. 115–129, 2012.

[76] Q. Zheng and F. Wu, "Nonlinear H-Infinity Control Designs with Axisymmetric Spacecraft Control," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 3, pp. 850–859, May 2009.

[77] P. A. Parrilo, "Exploiting Structure in Sum of Squares Programs," in *IEEE Conference on Decision and Control*, 2003, pp. 4664–4669 Vol.5.

[78] D. Jacobson and D. Mayne, *Differential Dynamic Programming*. Elsevier, 1970.

[79] C. G. Atkeson and B. J. Stephens, "Random Sampling of States in Dynamic Programming," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 38, no. 4, pp. 924–929, 2008.

[80] J. Morimoto, G. Zeglin, and C. G. Atkeson, "Minimax Differential Dynamic Programming: Application to a Biped Walking Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2003, pp. 1927–1932 vol.2.

[81] E. A. Theodorou, J. Buchli, and S. Schaal, "A Generalized Path Integral Control Approach to Reinforcement Learning," no. 11, pp. 3137–3181, 2010.

[82] E. Theodorou, J. Buchli, and S. Schaal, "Reinforcement Learning of Motor Skills in High Dimensions: a Path Integral Approach," in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 2397–2403.

[83] F. Stulp and O. Sigaud, "Path Integral Policy Improvement with Covariance Matrix Adaptation," in *International Conference on Machine Learning*, 2012.

[84] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback Motion Planning via Sums-of-Squares Verification," *International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[85] A. Majumdar, A. A. Ahmadi, and R. Tedrake, "Control Design Along Trajectories with Sums of Squares Programming," in *IEEE International Conference on Robotics and Automation*, May 2013, pp. 4054–4061.

[86] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[87] I. M. Gelfand and S. V. Fomin, *Calculus of Variations*, 1st ed. Courier Dover Publications, 2000.

[88] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, 2nd ed. Wiley, Apr. 2008.

[89] D. E. Kirk, *Optimal Control Theory: an Introduction*, 1st ed. Courier Dover Publications, 2012.

[90] A. Mehlmann, *Applied Differential Games*, 1st ed. Plenum Press, 1988.

[91] V. G. Boltyansky, "Robust Maximum Principle in Minimax Control," *International Journal of Control*, vol. 72, no. 4, pp. 305–314, 1999.

[92] V. Boltyanski, "Optimization and Robustness," *Optimization*, vol. 54, no. 4-5, pp. 369–376, 2005.

[93] V. G. Boltyanski and A. Poznyak, *The Robust Maximum Principle: Theory and Applications*, 1st ed., ser. Systems & Control: Foundations & Applications. Birkhäuser, Nov. 2011.

[94] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed., ser. Advances in Design and Control. Society for Industrial and Applied Mathematics, 2009.

[95] D. Zwillinger, *Handbook of Differential Equations*, 3rd ed. Gulf Professional Publishing, 1998.

[96] P. T. Boggs and J. W. Tolle, "Sequential Quadratic Programming," *Acta Numerica*, vol. 4, pp. 1–51, Jan. 1995.

[97] P. E. Gill, W. Murray, and M. Saunders, "SNOPT: an SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

[98] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A Unified Framework for the Numerical Solution of Optimal Control Problems Using Pseudospectral Methods," *Automatica*, vol. 46, no. 11, pp. 1843–1851, 2010.

[99] E. Meijering, "A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing," *Proceedings of the IEEE*, vol. 90, no. 3, pp. 319–342, Mar. 2002.

[100] R. Askey, *Orthogonal Polynomials and Special Functions*, 1st ed. Society for Industrial and Applied Mathematics, 1975.

[101] A. Quarteroni, F. Saleri, and R. Sacco, *Numerical Mathematics*, 1st ed. Springer, 2007.

[102] S. Subchan and R. Zbikowski, *Computational Optimal Control: Tools and Practice*, 1st ed. Wiley, 2009.

[103] J. Wang, E. C. Whitman, and M. Stilman, "Whole-Body Trajectory Optimization for Humanoid Falling," in *American Control Conference*, 2012, pp. 4837–4842.

[104] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained Model Predictive Control: Stability and Optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, June 2000.

[105] R. Franz, M. Milam, and J. Hauser, "Applied Receding Horizon Control of the Caltech Ducted Fan," in *American Control Conference*. IEEE, 2002, pp. 3735–3740 vol.5.

[106] M. Diehl, H. J. Ferreau, and N. Haverbeke, "Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation," in *Nonlinear Model Predictive Control: Towards New Challenging Applications*. London: Springer, 2011, pp. 391–417.

[107] W. Kwon and S. Han, *Receding Horizon Controls*, 1st ed. Springer, 2005.

[108] L. Magni and R. Sepulchre, "Stability Margins of Nonlinear Receding-Horizon Control via Inverse Optimality," *Systems & Control Letters*, vol. 32, no. 4, pp. 241–245, Dec. 1997.

[109] H. Chen, C. W. Scherer, and F. Allgöwer, "A Game Theoretic Approach to Nonlinear Robust Receding Horizon Control of Constrained Systems," in *American Control Conference*, June 1997, pp. 3073–3077 vol.5.

[110] R. Blauwkamp and T. Basar, "A Receding-Horizon Approach to Robust Output Feedback Control for Nonlinear Systems," in *IEEE Conference on Decision and Control*. IEEE, 1999, pp. 4879–4884 vol.5.

[111] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgöwer, "Robust Receding Horizon Control for Nonlinear Discrete-Time Systems," in *IFAC World Congress*, 2002.

[112] L. Magni, H. Nijmeijer, and A. J. van der Schaft, "A Receding-Horizon Approach to the Nonlinear H-Infinity Control Problem," *Automatica*, vol. 37, no. 3, pp. 429–435, 2001.

[113] S. Yu, C. Maier, H. Chen, and F. Allgöwer, "Terminal Set of Min-Max Model Predictive Control with Guaranteed L2 Performance," in *IEEE Conference on Decision and Control*, 2012, pp. 3264–3269.

[114] A. Gautam, Y.-C. Chu, and Y. C. Soh, "Robust H-Infinity Receding Horizon Control for a Class of Coordinated Control Problems Involving Dynamically Decoupled Subsystems," *IEEE Transactions on Automatic Control*, vol. 59, no. 1, pp. 134–149, Jan. 2014.

[115] M. Cutler, T. J. Walsh, and J. P. How, "Real-World Reinforcement Learning via Multifidelity Simulators," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 655–671, June 2015.

[116] A. G. Cunningham, E. Galceran, R. M. Eustice, and E. Olson, "MPDM: Multipolicy Decision-Making in Dynamic, Uncertain Environments for Autonomous Driving," in *IEEE International Conference on Robotics and Automation*, 2015.

[117] J.-S. Li and N. Khaneja, "Ensemble Control of Bloch Equations," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 528–536, Mar. 2009.

[118] J. Ruths, A. Zlotnik, and J.-S. Li, "Convergence of a Pseudospectral Method for Optimal Control of Complex Dynamical Systems," in *IEEE Conference on Decision and Control*, Dec. 2011, pp. 5553–5558.

[119] A. Becker and T. Bretl, "Approximate Steering of a Unicycle Under Bounded Model Perturbation Using Ensemble Control," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 580–591, June 2012.

[120] A. Varga, "Optimal Output Feedback Control: a Multi-Model Approach," in *IEEE International Symposium on Computer-Aided Control System Design*, 1996, pp. 327–332.

[121] A. Poznyak, T. Duncan, B. Pasik-Duncan, and V. G. Boltyanski, "Robust Maximum Principle for Multi-Model LQ-Problem," *International Journal of Control*, vol. 75, no. 15, pp. 1170–1177, 2002.

[122] A. Poznyak, F. J. Bejarano, and L. Fridman, "Numerical Method for Weights Adjustment in Minimax Multi-Model LQ-Control," *Optimal Control Applications and Methods*, vol. 28, no. 4, pp. 289–300, 2007.

[123] F. A. Miranda, F. Castaños, and A. Poznyak, "Min-Max Piecewise Constant Optimal Control for Multi-Model Linear Systems," *IMA Journal of Mathematical Control and Information*, 2015.

[124] V. Azhmyakov, V. G. Boltyanski, and A. Poznyak, "On the Dynamic Programming Approach to Multi-Model Robust Optimal Control Problems," in *American Control Conference*, June 2008, pp. 4468–4473.

[125] E. C. Whitman and C. G. Atkeson, "Multiple Model Robust Dynamic Programming," in *American Control Conference*, June 2012, pp. 5998–6004.

[126] M. McNaughton, "CASTRO: Robust Nonlinear Trajectory Optimization Using Multiple Models," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 177–182.

[127] C. G. Atkeson, "Efficient Robust Policy Optimization," in *American Control Conference*, June 2012, pp. 5220–5227.

[128] M. D. S. Aliyu, "Guaranteed Cost Control of Uncertain Nonlinear Systems: a Minimax Approach," in *IEEE Conference on Decision and Control*. IEEE, 1999, pp. 2485–2490 vol.3.

[129] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed. Dover Publications, 2001.

[130] J. Shen, T. Tang, and L.-L. Wang, *Spectral Methods: Algorithms, Analysis and Applications*, 1st ed., ser. Springer Series in Computational Mathematics. Springer, 2011, vol. 41.

[131] G. D. Konidaris, S. Osentoski, and P. S. Thomas, "Value Function Approximation in Reinforcement Learning Using the Fourier Basis," in *AAAI Conference on Artificial Intelligence*, Aug. 2011, pp. 380–385.

[132] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, Sept. 1999.

[133] M. C. Ferris and K. Sinapiromsaran, "Formulating and Solving Nonlinear Programs as Mixed Complementarity Problems," in *Optimization*, V. Nguyen, J.-J. Strodiot, and P. Tossings, Eds. Springer, 2000, pp. 132–148.

[134] R. H. Byrd, J. Nocedal, and R. A. Waltz, "KNITRO: an Integrated Package for Nonlinear Optimization," in *Nonconvex Optimization and Its Applications*, G. Pillo and M. Roma, Eds. Springer US, 2006, pp. 35–59.

[135] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. CRC Press, Mar. 1994.

[136] J. Peters and S. Schaal, "Learning to Control in Operational Space," *International Journal of Robotics Research*, vol. 27, no. 2, pp. 197–212, 2008.

[137] O. Khatib, "A Unified Approach for Motion and Force Control of Robot Manipulators: the Operational Space Formulation," *IEEE Journal of Robotics and Automation*, vol. 3, no. 1, pp. 43–53, Feb. 1987.

[138] M. Posa, C. Cantu, and R. Tedrake, "A Direct Method for Trajectory Optimization of Rigid Bodies Through Contact," *International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[139] M. Anitescu, "On Using the Elastic Mode in Nonlinear Programming Approaches to Mathematical Programs with Complementarity Constraints," *SIAM Journal on Optimization*, vol. 15, no. 4, pp. 1203–1236, 2005.

[140] S. M. LaValle, *Planning Algorithms*, 1st ed. Cambridge University Press, 2006.

[141] I. Mordatch and E. Todorov, "Combining the benefits of function approximation and trajectory optimization," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[142] C. Liu and C. G. Atkeson, "Standing Balance Control Using a Trajectory Library," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 3031–3036.

[143] J. Von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[144] L. J. Savage, *The Foundations of Statistics*. Courier Corporation, 1972.

[145] D. Kahneman and A. Tversky, "Prospect Theory: an Analysis of Decision Under Risk," *Econometrica: Journal of the Econometric Society*, pp. 263–291, 1979.