Title
Robust Model Predictive Control with Data-Driven Learning

Author
Bujarbaruah, Monimoy

Publication Date
2022

Peer reviewed|Thesis/dissertation

Robust Model Predictive Control with Data-Driven Learning

by

Monimoy Bujarbaruah

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Francesco Borrelli, Chair
Associate Professor Koushil Sreenath
Associate Professor Anil Aswani

Summer 2022

Robust Model Predictive Control with Data-Driven Learning

Abstract

Robust Model Predictive Control with Data-Driven Learning

by

Monimoy Bujarbaruah

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In the design of robust Model Predictive Control (MPC) algorithms, data can be used for primarily two purposes: (A) shrinking the feasible domain of the system uncertainty, and (B) enlarging the safe operating region of the system. In modern literature (A) is often referred to with model learning, or model adaptation, and (B) can be interpreted as using data to learn the model of the surrounding agents in the environment, or to learn environment constraints. Both (A) and (B) can enlarge the region of attraction of the MPC policy and improve its performance measured in terms of the closed-loop cost. However, the majority of the existing MPC algorithms that tackle (A) and (B) suffer from at least one the following deficiencies: (i) do not provide closed-loop guarantees of feasibility and stability, (ii) present conservative behavior as a result of over-approximation of the system uncertainty, (iii) are computationally expensive during online control synthesis, and (iv) cannot simultaneously handle system and environment constraint uncertainty for safe policy design.

In this dissertation, we present a unified framework to systematically incorporate data-driven learning in robust MPC design for linear dynamical systems. The proposed algorithms in the dissertation provide closed-loop guarantees, reduce conservatism in control design, and are computationally efficient and amenable for real-time implementation. The dissertation is divided into three parts where we focus on three aspects of learning during control design: model learning, disturbance distribution support learning, and environment constraint learning. Model learning and disturbance distribution support learning are instances of problem type (A), and environment constraint learning is an instance of problem type (B).

In the first part of the dissertation we consider model learning in linear time-invariant (LTI) and linear parameter-varying (LPV) systems where a reduction in the controller's conservatism is obtained by coupling novel ways of incorporating model learning in MPC with novel ways of robustifying the imposed constraints in MPC. We consider both parametric and non-parametric representation of the model uncertainty and present adaptive MPC algorithms that ensure robust satisfaction of the imposed state and input constraints.

In the second part we focus on learning the support of an additive disturbance's distribution. We consider the case when the disturbance belongs to the class of parametric distributions, and construct estimates of its unknown support via the confidence intervals of the underlying parameters. Robust MPC design with these learned supports can ensure satisfaction of the imposed constraints with any user-specified probability, while lowering conservatism by avoiding large outer-approximations of the true support.

Finally in the third part, we focus on learning unknown environment constraints imposed in the MPC optimization problem. We present a machine learning based algorithm to learn approximate constraint sets and validate their safety with samples of trajectory data. We prove that satisfying these approximated constraints with a robust MPC can guarantee probabilistic satisfaction of the actual constraints in closed-loop. The value of this probability can be chosen based on the desired trade-off between safety and performance of the controller.

We conclude the dissertation by presenting two applications where the proposed theory has been successfully tested. The first is for a robotic manipulator learning to play the cup-and-ball game, where we learn the support of a position measuring camera's measurement noise distribution from data and enable the robotic manipulator to play the game successfully using noisy camera feedback. For this case we present both high-fidelity simulation and experimental validations. The second application is for collaborative robotics, where we apply the concept of constraint learning in a decentralized collaborative robotic transportation scenario with partially known environment information to develop an obstacle avoidance algorithm. The algorithm allows the robots to adaptively assume leader-follower roles in the task while learning and avoiding unknown obstacles in their proximity.

To my family

Contents

List of Figures

List of Tables

Acknowledgments

Firstly, I wish to thank professor Francesco Borrelli for being an amazing advisor, mentor and a friend throughout my PhD. Your faith in me over these six years has taught me to be confident and trustful in my own abilities. Joining MPC lab was certainly one of the best decisions I have made in my life. I would also like to thank prof. Koushil Sreenath and prof. Anil Aswani for being in my dissertation committee. It is after taking a graduate class with prof. Aswani in my first year that I started my work on adaptive model predictive control, which is now an integral part of this thesis. Prof. Sreenath's feedback on my work on collaborative robotics has been extremely constructive and educative.

A special thanks goes to Xiaojing Zhang, Ugo Rosolia, Yvonne Stürz and Nitin Kapania for being inspirational peers and mentors. My research could not have been where it is, and I would not have had a thesis I could be satisfied with, if not for all those hours you spent helping and critiquing my work. I learned to think twice about all my research ideas and question each step, thanks to the value of diligence instilled by you all.

I'd be remiss if I don't acknowledge the incredible people back from my college days at IIT Bombay whom I still share deep and meaningful friendships with, including but not limited to Vishal Chourasiya, Chintan Dhruv, Atul Yadav, Abhijeet Dhiman, Avijit Singh, Saurabh Goel, Arpit Gupta and Ajit Singh. Of course, I now have to reserve a sizeable section here to thank all the friends I've been blessed with since I moved to Berkeley. So here we go: first of all, thanks to my lab mates Jon Gonzales, Edward Zhu, Hotae Lee, Siddharth Nair, Roya Firoozi, Charlott Vallon, Eric Choi, and Xu Shen. I have special memories with each of you and you all have been not only my colleagues, but great friends throughout my PhD. Thanks to Ugo Rosolia for always engaging in profound conversations on sorting out the priorities of one's life and learning to value each moment with family and loved ones. Thanks to Tony Zheng, for being an awesome research collaborator, friend, and a chef. Thanks to Vijay Govindarajan for being that role model who I could rely on for prudent life advice. Thanks to Yeojun Kim and Minyoung Chun for happily hosting and feeding me, and letting me binge watch Netflix for hours at your place. Thanks to Conrad Holda for never forgetting to remind me about the greatness of America (:P). Thanks to Saman Fahandezh-saadi for spending countless hours trying to turn me vegan and help me lead a healthy lifestyle. Thanks to Khalid Sefraoui for never trusting my healthy eating resolutions and reliably being there to see me order a burger and a beer shamelessly. Thanks to Eshetu Dejene and Logman Arja for being the two best "refugees" I've ever hosted in my apartment. You both cooked and fed me for months for free, and I still have no idea why! Thanks to Lars Svensson for being a great research collaborator and my polar Norway trip buddy. We did not see Aurora Borealis in 2020, but I am sure we will, even if it means freezing to death or being eaten by a polar bear in Svalbard. Thanks to Dimitris Papadimitriou for being there dissuading me from spending late nights in the lab and pushing me to "go out there". Thanks to Tony Kelman for tagging me along to countless concert nights in the Bay Area and that Iceland trip in 2018. Thanks to Yiwen Liao for never saying no when I suggested going for some lavish dinner, sometimes multiple times a week. Thanks to Sabrina Hussien

for listening to all my weird thoughts all the time and trying to discern some meaning out of them. Thanks to Emmanuel Sin and Galaxy Yin for reasons I should not publicly disclose. Our friend Barry will not appreciate that. Thanks to Eddie Kim for the warmly dinners and intriguing political discussions at your place. Thanks to Ivo Batkovic for being my licorice supplier from Scandinavia. Thanks to Prithvi Akella for being my role model in socializing with people. Thanks to Paolo Micalizzi for bringing in the much needed aura of nonchalance to our everyday lives and always sincerely playing the "will you do (something disturbing) for a billion dollars?" game and never letting me be the only one saying yes. And thanks to Nikita Tugarin for being who you are.

A special gesture of gratitude here goes to Akhil Shetty, Łukasz Langer, Ugo Rosolia, Jacopo Guanetti, Hannibal Birhane, Sabrina Hussien, Vijay Govindarajan, Abhinav Bhattacharyya, Tony Zheng, and Mrinal Kalita. Each and every time we talk, I got better insights into my own thoughts and slowly learned to investigate them with curiosity and empathy, instead of harsh self judgement. Having had people like you, who come from myriad different backgrounds, listening to me and providing your invaluable perspectives, made me realize what deep friendships are for. Today I believe I am a genuinely equanimous being, thanks to all your empathy, compassion and kindness directed at me when I was the exact opposite. Thanks to Akhil, I started Vipassana and Metta practices a couple of years back, which have been life changing to say the least. Also life changing were the glasses of port wine, chunks of panettone and bowls of mascarpone I devoured with Jacopo.

And finally, thanks to my family for your unwavering support and encouragement during my pursuit of life aspirations, while refraining from imposing any expectations and rules whatsoever on my beliefs and lifestyle. I learned to think independently and grow as an individual, thanks to your everlasting faith in me. You've been akin to a group of amazing friends, and I can't express how lucky I am to have had your support with me not only during this PhD, but throughout my life.

Chapter 1

Introduction

In the past decade there has been a renewed interest in data-driven methods for safe control design for uncertain systems under state and input constraints[1, 2, 3]. The uncertainty in these systems can be typically attributed to two factors: (i) model uncertainty (e.g., modeling mismatch and inaccuracies), and (ii) exogenous disturbances[1]. For such uncertain systems subject to state and input constraints, robust Model Predictive Control (MPC) [4, 5, 6] is a commonly used approach for ensuring robust constraint satisfaction. In the design of robust MPC algorithms, data can be used for primarily two purposes: (A) shrinking the feasible domain of the system uncertainty (e.g., model learning, disturbance bound learning), and (B) enlarging the safe operating region of the system (e.g, constraint learning). Both (A) and (B) can enlarge the region of attraction of the MPC policy and improve its performance measured in terms of the closed-loop cost.

In this dissertation, we present a set of algorithms that utilize data-driven learning in robust MPC design in order to lower controller conservatism and improve its performance during operation. We focus on linear systems and three main components for learning, namely: model learning, disturbance distribution support learning, and environment constraint learning. The existing work in each of these three cases, their shortcomings, and the contributions in this dissertation are summarized below:

Model Learning

Data-driven learning has been utilized to lower the conservatism of an MPC by learning the domain of the model uncertainty. Works such as [7, 8, 9] use a Gaussian Process (GP) regression [10, 11] for online model learning and adaptation, allowing the room for violations of the imposed constraints with a certain user-specified probability. However, they provide no theoretical bounds on the rate of constraint violations by the closed-loop system over time. Adaptive MPC works such as [12, 13, 14] learn and update the feasible parameters of a parametric model uncertainty and provide closed-loop guarantees of robust constraint

[1]We primarily consider the case of perfect state measurements in this dissertation during control design. Therefore the contribution of measurement noise is omitted in the system uncertainty quantification.

satisfaction and stability. However these methods can incur high online computational expenses similar to polytopic tube MPC algorithms [15, 16, 17, 18, 19], primarily due to an increase in the number of constraints in the MPC problem. Balancing this trade-off between computational complexity and controller conservatism is a key aspect during robust MPC design for this class of systems [2].

In this dissertation we build a set of adaptive MPC algorithms motivated by the works of [20, 21, 22, 12, 13, 8]. We learn both parametric and non-parametric model uncertainties using set membership estimation [23] and graph learning algorithms [24, 25]. We also develop two novel algorithms for robust MPC design for linear systems under both additive and multiplicative uncertainties, which are used for control design while learning linear parameter varying (LPV) models. These two robust MPC algorithms can obtain an improved computational complexity vs conservatism trade-off over methods such as [15, 16, 19, 26, 27], as we show in Chapter 3 and Chapter 4 with detailed numerical examples. The algorithms presented in this dissertation have appeared in [28, 29, 30, 31, 30, 32, 33].

Disturbance Distribution Support Learning

If the support of the disturbance distribution is not exactly known, using over-approximations results in conservative controller behavior [5]. This motivates learning the disturbance support over time using collected data from the system. In such cases, it is necessary to allow the possibility of *failure*, i.e., violation of imposed constraints by the MPC, as the learned support may not entirely contain the true support of the disturbance. To the best of our knowledge, the only data-driven approach for learning the support of the disturbance distribution during robust MPC design is presented in [34], which constructs estimated disturbance support sets offline using the scenario approach [35, Chapter 12]. The approach in [34] involves solving a scenario program with potentially large number of samples, which is computationally expensive. Moreover, the rate of constraint violation in closed-loop, i.e., failures, is dependent on the number of disturbance samples available offline for solving the scenario program. Thus, [34] is unable to satisfy a desired upper bound on failures at all times, since the required number of samples could be unavailable during operation.

In this dissertation we present an iterative algorithm called *Learning Robustness with Bounded Failure* (LRBF) [36] which learns the supports of additive disturbances in a linear time-invariant model by utilizing confidence intervals of the parameters that define the parametric distributions of the disturbances. Unlike methods such as [37, 34], we can guarantee a user-specified upper bound on the constraint violation probability by a robust MPC using these estimated supports, from the very start of the control task. As more iteration data is collected, the estimated support approaches the true one, and thus lowering the probability of constraint violations. We use the LRBF algorithm to learn the support of a camera's position measurement noise distribution and design an output feedback robust optimal controller, enabling a robotic manipulator to learn to play the cup-and-ball game. The catching

[2] See Chapter 2 for a detailed discussion.

of the ball by the manipulator improves as the approximated support of the camera noise is refined with data, thus improving the accuracy of the catching controller.

Constraint Learning

Data-driven methods are also used to enlarge the ROA and improve the controller performance by learning unknown environment constraints. A set of such algorithms in MPC design can be found in [38, 39, 40]. However, the systems considered in these aforementioned works are deterministic, i.e., free of uncertainty. To the best of our knowledge, the literature on data-driven controller design in the presence of uncertainties in *both* the system and the environment constraint set is rather limited.

In this dissertation we develop an iterative algorithm called *Iterative Constraint Learning* (ICL) for environment constraint learning [41] during robust MPC design, where approximations of unknown environment constraints are learned from trajectory data using kernel support vector machines [42, Chapter 12]. The violation probability of the true unknown environment constraints is ensured bounded by a user-specified value. The higher this probability, the lower the incurred average closed-loop cost by the system, thus highlighting a safety vs performance trade-off, which can be decided by the user. We further extend this concept of constraint learning to applications in decentralized collaborative robotics [43], where two robots adaptively assume leader-follower roles during a collaborative object transportation task and improve their obstacle avoiding MPC planners by inferring unknown obstacle information in their proximity.

1.1 Outline

We now present an outline of this dissertation in the following section. Chapters 2-8 constitute the theoretical foundations of the dissertation. In Chapters 9-10 we present two application cases for the proposed theoretical work.

Chapter 2: Background on Robust MPC

In this chapter we present a background on robust MPC algorithms. We highlight the computational complexity vs conservatism trade-off that exists in the design of these algorithms, and thus motivate the need for (i) two novel robust MPC algorithms that we propose in Chapter 3 and Chapter 4, and (ii) data-driven learning in conjunction with robust MPC.

Chapter 3: A Simple Robust MPC for LPV Systems

In this chapter we propose a simple algorithm for robust MPC design for LPV systems. This algorithm uses two bounding strategies for system uncertainty as follows: (i) a worst-case bound for constraint tightening along the prediction horizon, which is computed by lumping

up the contribution of matrix uncertainties and the additive disturbances into one "net-additive term", and (*ii*) a terminal set without over-approximating the system uncertainty. We demonstrate with numerical examples that using these two bounds with an adaptive horizon strategy leads to an MPC which obtains an improved conservatism vs computational speed trade-off over existing approaches, such as [15, 27].

Chapter 4: Robust MPC with Optimization-Based Constraint Tightening

In this chapter we propose an optimization-based constraint tightening strategy for designing a robust MPC algorithm. The constraint tightenings in the control synthesis problem are a function of the predicted nominal states and inputs, i.e., the decision variables. This lowers the conservatism in the proposed control design approach by avoiding worst-case bounds as used in the "net-additive" uncertainty term in Chapter 3. The resultant MPC problem is computationally efficient, and obtains an improved ROA over the robust MPC in Chapter 3 for our considered simulation cases.

Chapter 5: Learning Non-Parametric Model Uncertainty in Robust MPC

In this chapter we propose our first robust adaptive MPC algorithm, where the model uncertainty is additive and state dependent. We assume the uncertainty is globally Lipschitz, with a known Lipschitz constant. We utilize a non-parametric recursive system identification strategy which identifies the graph of the uncertainty from data using its Lipschitz property. The identification is successively refined with recorded data. The bounds of the uncertainty evolution along the prediction horizon are obtained via the s-procedure [44, 45], and utilizing these bounds a robust MPC controller is designed. We demonstrate with numerical examples that the adaptation of system uncertainty using data shrinks these uncertainty bounds with time, thus lowering the conservatism of the controller.

Chapter 6: Learning Parametric Model Uncertainty in Robust MPC

In this chapter we propose a tractable adaptive MPC framework for linear systems that are subject to bounded additive uncertainty, which is composed of a disturbance, and an unknown, but bounded parametric offset. We learn and refine the feasible domain of this offset parameter using collected data via set membership methods. A robust MPC is then designed for all feasible offsets in the domain, which incurs lower conservatism as the offset domain is refined with time. We then extend this robust adaptive MPC design to LPV systems using the robust MPC proposed in Chapter 3.

Chapter 7: Learning Disturbance Distribution Supports in Robust MPC

In this chapter we present an approach to design an MPC controller for constrained linear time-invariant systems performing an iterative task, where the support of the additive disturbance is learned from data. We call our algorithm *Learning Robustness with Bounded Failure* (LRBF). We demonstrate that LRBF is able to learn the true support of the disturbance asymptotically, and thus lowers the controller's conservatism over approaches which use large outer approximations of the unknown disturbance support. Furthermore, while learning the support of the additive disturbance, we guarantee a user-specified upper bound on the probability of failure over all iterations.

Chapter 8: Learning Environment Constraints in Robust MPC

This chapter focuses on improving the closed-loop cost performance of a robust MPC controller, while satisfying the safety constraints imposed by the environment. Instead of relying on small inner approximations of the environment constraints, we learn them from collected system trajectories. For this, we use a classifier, which provides constraint violation-satisfaction flags at each timestep, given a recorded closed-loop trajectory. Using this flag information, we compute a constraint estimate set using standard nonlinear regression tools. The estimated set is verified using a randomized algorithm, which ensures that the MPC designed to satisfy the estimated sets robustly satisfies the true (i.e., unknown) constraints with a user-specified probability.

Chapter 9: Playing Cup-and-Ball: An Application of LRBF

In this chapter we demonstrate a fully physics driven model-based hybrid approach for control design in order for a robotic manipulator to learn to play the cup-and-ball game. The manipulator uses noisy measurements from a camera to obtain the ball's position information. The camera noise support is refined with data using the tools presented in Chapter 7. We demonstrate that the aforementioned use of data in designing a feedback controller for the manipulator improves its catching capabilities with time. Furthermore, the robust optimal control problem solved for control synthesis in this case is with noisy output feedback [46], and therefore an observer design is also involved before control synthesis.

Chapter 10: Decentralized Robotic Collaboration: An Application of Constraint Learning

In this chapter we consider the task of two robots collaboratively transporting an object through an obstacle prone environment without any explicit communication between them. This implies that the local environment information and the control actions are not shared between the robots. We solve the control design problem in a decentralized manner by using

a leader-follower strategy, with the leader robot using an MPC and the follower using a simple controller, known to the leader. Motivated by the tools of Chapter 8, the leader builds a map of its unknown obstacles, i.e., constraints in its proximity, purely relying upon its own estimates and/or haptic feedback from the follower. Thus, the leader's MPC policy improves as more data is collected along the task. With numerical simulations we demonstrate that this proposed obstacle inference/learning approach allows the robots to safely complete the transportation task, while adaptively switching leader-follower roles based on the inferred environment information.

Chapter 2

Technical Background on Robust MPC

2.1 Invariant Sets for Nominal Systems

In this section we consider deterministic linear time-invariant systems. The following definitions of invariant sets will be useful for obtaining feasibility guarantees of robust MPC controllers. Let $\mathcal{X}$ and $\mathcal{U}$ denote the state and input constraint sets, respectively.

Definition 2.1 (Positive Invariant Set) *Given a policy $\pi(\cdot)$, a set $\mathcal{O} \subseteq \mathcal{X}$ is said to be a positive invariant set for the deterministic autonomous system $x_{t+1} = Ax_t + B\pi(x_t)$, if*

$$x \in \mathcal{O} \rightarrow Ax + B\pi(x) \in \mathcal{O}.$$

Definition 2.2 (Maximal Positive Invariant Set) *Given a policy $\pi(\cdot)$, a set $\mathcal{O} \subseteq \mathcal{X}$ is said to be the maximal positive invariant set for the deterministic autonomous system $x_{t+1} = Ax_t + B\pi(x_t)$, if $\mathcal{O}$ is a positive invariant set and it contains all the positive invariant sets contained in $\mathcal{X}$.*

Definition 2.3 (Control Invariant Set) *A set $\mathcal{C} \subseteq \mathcal{X}$ is said to be a control invariant set for the deterministic system $x_{t+1} = Ax_t + Bu_t$, with $u_t \in \mathcal{U}$ if*

$$x \in \mathcal{C} \rightarrow \exists u \in \mathcal{U} : Ax + Bu \in \mathcal{C}.$$

2.2 Robust Invariant Sets, Robust Controllable and Reachable Sets

In this section, we consider uncertain linear time-invariant systems. The following definitions will be used subsequently to synthesize robust MPC algorithms. Recall, $\mathcal{X}$ and $\mathcal{U}$ denote the state and input constraint sets, respectively.

Definition 2.4 (Robust Positive Invariant Set) *Given a policy $\pi(\cdot)$, a set $\mathcal{O} \subseteq \mathcal{X}$ is said to be a robust positive invariant set for the uncertain autonomous system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, with $w_t \in \mathbb{W}$ if*

$$x \in \mathcal{O} \rightarrow Ax + B\pi(x) + w \in \mathcal{O}, \ \forall w \in \mathbb{W}.$$

Definition 2.5 (Maximal Robust Positive Invariant Set) *Given a policy $\pi(\cdot)$, a set $\mathcal{O} \subseteq \mathcal{X}$ is said to be the maximal robust positive invariant set for the uncertain autonomous system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, with $w_t \in \mathbb{W}$ if $\mathcal{O}$ is a robust positive invariant set and it contains all the robust positive invariant sets contained in $\mathcal{X}$.*

Definition 2.6 (Minimal Robust Positive Invariant Set) *Given a policy $\pi(\cdot)$, a set $\mathcal{O} \subseteq \mathcal{X}$ is said to be the minimal robust positive invariant set for the uncertain autonomous system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, with $w_t \in \mathbb{W}$ if $\mathcal{O}$ is a robust positive invariant set and it is contained in all the robust positive invariant sets contained in $\mathcal{X}$.*

Definition 2.7 (Robust Precursor Set) *Given a control policy $\pi(\cdot)$ and the closed-loop system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$ with $w_t \in \mathbb{W}$ for all $t \geq 0$, we denote the robust precursor set to the set $\mathcal{S}$ under a policy $\pi(\cdot)$ as*

$$\text{Pre}(\mathcal{S}, A, B, \mathbb{W}, \pi(\cdot)) = \{x \in \mathbb{R}^n : Ax + B\pi(x) + w \in \mathcal{S}, \forall w \in \mathbb{W}\}. \tag{2.1}$$

$\text{Pre}(\mathcal{S}, A, B, \mathbb{W}, \pi(\cdot))$ *defines the set of states of the system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, which evolve into the target set $\mathcal{S}$ in one timestep for all $w_t \in \mathbb{W}$.*

Definition 2.8 (N-Step Robust Controllable Set) *Given a control policy $\pi(\cdot)$ and the closed-loop system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, we recursively define the N-Step Robust Controllable set to the set $\mathcal{S}$ as*

$$\begin{aligned} &\mathcal{C}_{t\rightarrow t+k+1}(\mathcal{S}) = \text{Pre}(\mathcal{C}_{t\rightarrow t+k}(\mathcal{S}), A, B, \mathbb{W}, \pi(\cdot)) \cap \mathcal{X}, \\ &\text{with } \mathcal{C}_{t\rightarrow t}(\mathcal{S}) = \mathcal{S}, \end{aligned}$$

for $k \in \{0, 1, \ldots, N-1\}$.

The N-Step Robust Controllable set $\mathcal{C}_{t\rightarrow t+N}(\mathcal{S})$ collects the states satisfying the state constraints which can be steered to the set $\mathcal{S}$ in N steps under the policy $\pi(\cdot)$, for all possible disturbance realizations.

Definition 2.9 (Robust Successor Set) *Given a control policy $\pi(\cdot)$ and the closed-loop system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$, we denote the robust successor set from the set $\mathcal{S}$ as*

$$\text{Succ}(\mathcal{S}, \mathbb{W}, \pi(\cdot)) = \{x_{t+1} \in \mathbb{R}^n : \exists x_t \in \mathcal{S}, \exists w_t \in \mathbb{W} \text{ such that } x_{t+1} = Ax_k + B\pi(x_t) + w_t\}.$$

Given the initial state x_t, the robust successor set $\text{Succ}(x_t, \mathbb{W})$ collects the states that the uncertain autonomous system may reach in one timestep.

2.3 Additional Definitions

Definition 2.10 (Lyapunov Function) *Consider the equilibrium point $x = 0$ of an autonomous system $x_{t+1} = f(x_t)$. Let $\Omega \subset \mathbb{R}^n$ be a closed and bounded set containing the origin. Assume there exists a function $V : \mathbb{R}^n \to \mathbb{R}$ continuous at the origin, finite for every $x \in \Omega$, and such that*

$$V(0) = 0 \text{ and } V(x) > 0, \forall x \in \Omega \setminus \{0\}, \tag{2.2a}$$
$$V(f(x)) - V(x) \leq 0. \tag{2.2b}$$

Then $x = 0$ is asymptotically stable in the sense of Lyapunov on Ω, and the function $V(\cdot)$ satisfying conditions (2.2) is called a Lyapunov Function.

Definition 2.11 (Control Lyapunov Function) *Consider system $x_{t+1} = f(x_t, u_t)$ subject to the state and input constraint, $x_t \in \mathcal{X}$ and $u_t \in \mathcal{U}$. Assume that $\mathcal{S}$ is a control invariant set and $\ell(x, u)$ is the stage cost of the control problem. Then, the function $Q : \mathbb{R}^n \to \mathbb{R}$ is a control Lyapunov function over the set $\mathcal{S}$ if*

$$\forall x \in \mathcal{S}, \quad \min_{u \in \mathcal{U}, f(x,u) \in \mathcal{S}} \left[\ell(x, u) + Q(f(x, u)) - Q(x) \right] \leq 0.$$

Definition 2.12 (Class–$\mathcal{K}$ Function) *A continuous function $\alpha : [0, a) \to [0, \infty)$ is called a class-$\mathcal{K}$ function if it is strictly increasing in its domain and if $\alpha(0) = 0$. The class-$\mathcal{K}$ function belongs to class-$\mathcal{K}_\infty$ if $a = \infty$ and $\lim_{r \to \infty} \alpha(r) = \infty$.*

Definition 2.13 (Class-$\mathcal{KL}$ Function) *A continuous function $\beta : [0, a) \times [0, \infty) \mapsto [0, \infty)$ is called a class-$\mathcal{KL}$ function if for each fixed s, the function $\beta(r, s)$ belongs to class-$\mathcal{K}$, and for each fixed r, (i) the value $\beta(r, s)$ is decreasing w.r.t. s and (ii) $\beta(r, s) \to 0$ for $s \to \infty$.*

Definition 2.14 (Lipschitz Function) *A real valued function $\alpha : [a, b] \mapsto \mathbb{R}$ is called Lipschitz with a Lipschitz constant L, if for all $x, y \in [a, b]$, we have $\|\alpha(x) - \alpha(y)\| \leq L\|x - y\|$, where $\| \cdot \|$ denotes the norm of a vector.*

Definition 2.15 (Minkowski Sum) *The Minkowski sum of two sets $\mathcal{P}, \mathcal{Q} \subseteq \mathbb{R}^n$ is defined as:*

$$\mathcal{P} \oplus \mathcal{Q} = \{x + y : x \in \mathcal{P}, y \in \mathcal{Q}\}.$$

Definition 2.16 (Pontryagin Difference) *The Pontryagin difference between two sets $\mathcal{P}, \mathcal{Q} \subseteq \mathbb{R}^n$ is defined as:*

$$\mathcal{P} \ominus \mathcal{Q} = \{x : x + q \in \mathcal{P}, \forall q \in \mathcal{Q}\}.$$

2.4 Notation

The following notations will be used throughout this dissertation. x^+ denotes one timestep updated value of x. The induced p-norm of any matrix A is given by $\|A\|_p = \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_p}$, where $\|\cdot\|_p$ is the p-norm of a vector. The set $K\mathcal{B}$ denotes the set of elements obtained from multiplying each element in the set $\mathcal{B}$ with K, i.e., $K\mathcal{B} = \{x : x = bK, b \in \mathcal{B}\}$. The sign $u \geq v$ between two vectors u, v denotes element-wise inequality. $\text{conv}(X, Y, \ldots, Z)$ denotes the set of matrices that can be written as a convex combination of the matrices $X, Y, \ldots, Z$. I_n denotes an identity matrix of dimension n and 1_n denotes a vector of ones of length n. The dual norm of any vector norm $\|x\|$ for a vector x is defined as $\|x\|_* = \sup_{\|v\|\leq 1}(v^\top x)$. The consistency property for any induced p-norm and vector q-norm is given by $\|Xy\|_q \leq \|X\|_p\|y\|_q$, for any $X \in \mathbb{R}^{d_1\times d_2}$ and $y \in \mathbb{R}^{d_2}$. The submultiplicativity property for any induced p-norm is given by $\|XY\|_p \leq \|X\|_p\|Y\|_p$. Inequality $X \geq 0$ for any matrix X denotes an element-wise inequality, whereas a conic inequality is denoted by $X \succeq 0$. $\mathbf{0}_n$ is a vector of n zeros and $\mathbf{0}_{m\times n}$ is an $m \times n$ matrix of zeros.

2.5 Robust MPC Problem Formulation

In this section we present an overview of robust MPC design for an uncertain nonlinear system given by:

$$x_{t+1} = f(x_t, u_t, w_t), \tag{2.3}$$

where $x_t \in \mathbb{R}^n$ is the state, $u_t \in \mathbb{R}^m$ is the input, and $w_t \in \mathbb{W} \subset \mathbb{R}^n$ is the disturbance lying on a compact support $\mathbb{W}$. The system is subject to state and input constraints

$$x_t \in \mathcal{X},\ u_t \in \mathcal{U},\ \forall t \geq 0, \tag{2.4}$$

where $\mathcal{X}$ and $\mathcal{U}$ are compact. We assume that dynamics $f(\cdot,\cdot,\cdot)$ is not known exactly, and we classify the associated uncertainty as follows:

Model Uncertainty: uncertainty in $f(\cdot,\cdot,0)$, which we denote as *model uncertainty*. Such model uncertainty typically arises due to unknown components in the physics of the model, e.g., unknown parameters such as mass, moment of inertia, etc.

Exogenous Disturbances: these are external influences in the system evolution, denoted by w_t. Although such external disturbances can have their own evolution dynamics, for the remainder of this dissertation we focus on the case where the disturbance samples are stochastic, and in particular i.i.d. in nature.

A nominal estimated model $\bar{f}(\cdot,\cdot,0)$ is used for control design. The model uncertainty in $f(\cdot,\cdot,0)$ can be represented either with a parametric or with a non-parametric representation. These are elaborated next. See Remark 2.3 for the corresponding details on the specific models considered subsequently in this dissertation.

Parametric Model Uncertainty

The model may contain some unknown parameters $\theta_t \in \mathbb{R}^p$ parametrizing $f_{\theta_t}(\cdot,\cdot,0)$, which are estimated during control design as $\bar{\theta}_t$. In this context the nominal model is $\bar{f}_{\bar{\theta}_t}(\cdot,\cdot,0)$. The measure of modeling error in this case is obtained from the error in parameter estimation, i.e.,

$$\Delta\theta_t^{\text{tr}} = \theta_t^{\text{tr}} - \bar{\theta}_t,$$

where the true parameter value θ_t^{tr}, and thus $\Delta\theta_t^{\text{tr}}$ is unknown, but is bounded. In this thesis, we assume that $\Delta\theta_t^{\text{tr}}$ lies in a known set $\mathcal{F}_t$ which is given by:

$$\mathcal{F}_t = \{\Delta\theta_t : \|\Delta\theta_t\| \leq \delta_t\}, \text{ with some known } \delta_t > 0. \tag{2.5}$$

Parametric models can be expressive, e.g., neural networks are instances of parametric models. Nonetheless, based on the specific application under consideration, non-parametric representation of uncertain models is also common in MPC literature.

Non-parametric Model Uncertainty

In the non-parametric uncertainty representation approach, the relationship between the estimated model $\bar{f}(\cdot,\cdot,0)$ and the true model $f(\cdot,\cdot,0)$ is expressed as:

$$f(x_t,u_t,0) = \bar{f}(x_t,u_t,0) + \Delta_{f,t}^{\text{tr}}(x_t,u_t),$$

where the *mismatch* function $\Delta_{f,t}^{\text{tr}}(x_t,u_t)$ is unknown and assumed bounded. The equivalent model error $f(\cdot,\cdot,0) - \bar{f}(\cdot,\cdot,0)$ for this approach is given by a set of functions $\mathcal{F}_t$. This set can be characterized by known set-valued maps as given by:

$$\mathcal{F}_t = \{\Delta_{f,t}(\cdot,\cdot) : \Delta_{f,t}(x_t,u_t) \in \mathcal{D}(x_t,u_t)\}, \tag{2.6}$$

where $\mathcal{D}(x,u)$ is a known bounded set of possible values at (x,u), e.g., confidence set obtained from a Gaussian Process (GP) regression [47]. This set in general is time-varying, i.e., $\mathcal{D}_t(x_t,u_t)$. But for simplicity of notations subsequently in Section 2.11 and Chapter 5, we omit the time index subscript.

The Robust MPC Problem

Let the MPC horizon be N. Let $x_{k|t}$ denote the predicted state at timestep k for any possible uncertainty realization, obtained by applying the sequence of predicted input policies $\{u_{t|t}, u_{t+1|t}(\cdot),\ldots,u_{k-1|t}(\cdot)\}$ to system (2.3), and $\{\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})\}$ denote the nominal state and corresponding input respectively. In general for a robust MPC synthesis, we consider

solving the following optimal control problem at each timestep t:

$$\min_{u_{t|t},u_{t+1|t}(\cdot),\ldots,u_{t+N-1|t}(\cdot)} \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})) + Q(\bar{x}_{t+N|t}) \tag{2.7a}$$

$$\text{s.t.,} \quad x_{k+1|t} = f(x_{k|t}, u_{k|t}(x_{k|t}), w_{k|t}), \tag{2.7b}$$

$$\bar{x}_{k+1|t} = \bar{f}(\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t}), 0), \tag{2.7c}$$

$$x_{k|t} \in \mathcal{X},\ u_{k|t}(x_{k|t}) \in \mathcal{U}, \tag{2.7d}$$

$$x_{t+N|t} \in \mathcal{X}_N, \tag{2.7e}$$

$$\forall w_{k|t} \in \mathbb{W},\ \forall \Delta\theta_t \in \mathcal{F}_t,\ \text{if model (2.5) is used, or}$$
$$\forall \Delta_{f,t} \in \mathcal{F}_t,\ \text{if model (2.6) is used,} \tag{2.7f}$$

$$\forall k = \{t, \ldots, t+N-1\}, \tag{2.7g}$$

$$x_{t|t} = x_t,\ \bar{x}_{t|t} = x_t, \tag{2.7h}$$

with $U_t(\cdot) = \{u_{t|t}, u_{t+1|t}(\cdot), \ldots, u_{t+N-1|t}(\cdot)\}$, and applying the optimal MPC policy

$$u_t^{\text{MPC}}(x_t) = u_{t|t}^{\star}, \tag{2.8}$$

to system (2.3) in closed-loop.

Remark 2.1 *Note that we have considered perfect state feedback in* (2.7h)*. Robust MPC synthesis with noisy output feedback is presented in [46, 48, 49], etc. Although we utilize the output feedback robust MPC of [46] in Chapter 9 of this thesis, we will primarily limit our focus to when* (2.7h) *holds.*

Note, the first input $u_{t|t}$ is not a policy, as the state x_t is known exactly. The objective is to minimize the cost associated with the nominal model (2.7c). Constraints (2.7d)-(2.7e) are satisfied robustly for all possible set of states reachable through any feasible choice of the true model evolution (2.7b), i.e., for all uncertainty in (2.7f), where $\mathcal{F}_t$ is given by (2.5) or (2.6) depending on parametric or non-parametric uncertainty representation in the model.

Challenges in Solving (2.7), Assumptions and Simplifications

There are three main challenges with solving (2.7), namely:

(C1) The state and input constraints are to be satisfied robustly under the presence of mismatch between the nominal and the true system models and all possible disturbances. In other words, (2.7d)-(2.7e)-(2.7f) need to be reformulated so that they can be fed to a numerical programming algorithm.

(C2) Optimizing over policies $\{u_{t|t}, u_{t+1|t}(\cdot), u_{t+2|t}(\cdot), \ldots\}$ in (2.7) involves an optimization over infinite dimensional function spaces. This in general is not computationally tractable [50, 6].

(C3) The feasibility of problem (2.7) is to be guaranteed robustly at all timesteps $t \geq 0$. That is,

$$x_t \in \mathcal{X}, u_t^{\text{MPC}}(x_t) \in \mathcal{U}, \ \forall w_t \in \mathbb{W}, \forall t \geq 0,$$

where $x_{t+1} = f(x_t, u_t^{\text{MPC}}(x_t), w_t)$. Furthermore, the closed-loop system is to be stabilized by the MPC law (2.8).

Remark 2.2 *Note that although we choose to show the minimization of the nominal cost in (2.7a), this cost can be chosen as the expected cost, or the worst-case cost. The choice of the cost function has no impact on tackling challenges (C1)-(C3). In fact, the set of specific algorithms discussed in Section 2.9 use a worst-case cost.*

To tackle challenge (C1), simplifying assumptions are made in MPC literature to the system (2.3). In this dissertation, we consider the following three type of simplified systems:

(M1) The system is linear time-invariant with an additive disturbance, i.e.,

$$x_{t+1} = Ax_t + Bu_t + w_t, \tag{2.9}$$

with known matrices A and B of appropriate dimensions. In this case the true disturbance free and the nominal models are the same, i.e., $f(\cdot,\cdot,0) = \bar{f}(\cdot,\cdot,0)$. That means there is no model uncertainty in the system; the only source of uncertainty is the disturbance w_t.

(M2) The system is linear parameter varying with unknown system matrices and an additive disturbance, i.e., system (2.3) simplifies to

$$x_{t+1} = (\bar{A} + \Delta_A)x_t + (\bar{B} + \Delta_B)u_t + w_t, \tag{2.10}$$

where we assume that A and B are unknown matrices with estimates $\bar{A}$ and $\bar{B}$ available to the control designer. In particular, we consider

$$A = \bar{A} + \Delta_A^{\text{tr}}, \ B = \bar{B} + \Delta_B^{\text{tr}}, \tag{2.11}$$

where the true parametric uncertainty matrices Δ_A^{tr} and Δ_B^{tr} are unknown and belong to convex and compact sets

$$\Delta_A^{\text{tr}} \in \mathcal{P}_A, \ \Delta_B^{\text{tr}} \in \mathcal{P}_B. \tag{2.12}$$

Furthermore, we consider that the sets $\mathcal{P}_A$ and $\mathcal{P}_B$ are convex hulls of known *vertex* matrices $\{\Delta_A^{(1)}, \Delta_A^{(2)}, \ldots, \Delta_A^{(n_a)}\}$ and $\{\Delta_B^{(1)}, \Delta_B^{(2)}, \ldots, \Delta_B^{(n_b)}\}$, with fixed $n_a, n_b > 0$:

$$\begin{aligned} \mathcal{P}_A &= \text{conv}(\Delta_A^{(1)}, \Delta_A^{(2)}, \ldots, \Delta_A^{(n_a)}), \\ \mathcal{P}_B &= \text{conv}(\Delta_B^{(1)}, \Delta_B^{(2)}, \ldots, \Delta_B^{(n_b)}). \end{aligned} \tag{2.13}$$

(M3) As a special instance of approximating the nonlinearities in the model in (2.3) as an additive model uncertainty, we consider a linear time-invariant system with a state dependent additive uncertainty, i.e., (2.3) simplifies to

$$x_{t+1} = Ax_t + Bu_t + d(x_t), \tag{2.14}$$

where the unknown, nonlinear function $d(x)$ is bounded over the state-space, i.e., $d(x) \in \mathcal{D}(x)$ for all $x \in \mathcal{X}$ for a compact set $\mathcal{D}(x)$.

Remark 2.3 *In this dissertation when we present algorithms for utilizing data to refine uncertainty in systems of the forms (M1)-(M3), we consider the following cases:*

- *For models of the form (M1), for learning an unknown support $\mathbb{W}$ of the disturbance w_t, we consider the distribution of w_t as a parametric distribution $\mathcal{P}_\theta$ [51] with unknown parameters θ, and the estimated supports $\hat{\mathbb{W}}_{\hat{\theta}}$ are obtained from the corresponding confidence intervals of the parameters. The associated algorithm is presented in Chapter 7.*

- *For models of the form (M2), we consider a specific instance with system matrices $(A(\theta_t^{\text{tr}}), B(\theta_t^{\text{tr}}))$ with an unknown parameter $\theta_t^{\text{tr}} \in \Theta_t$. More specifically,*

$$x_{t+1} = A(\theta_t^{\text{tr}})x_t + B(\theta_t^{\text{tr}})u_t + w_t, \;\; w_t \in \mathbb{W}, \tag{2.15}$$

where $\theta_t^{\text{tr}} \in \mathbb{R}^p$ is a time-varying parameter unknown to the control designer, which decides the values of the system matrices as:

$$(A(\theta_t^{\text{tr}}), B(\theta_t^{\text{tr}})) = (A_0, B_0) + \sum_{i=1}^{p} (A_i \theta_{i,t}^{\text{tr}}, B_i \theta_{i,t}^{\text{tr}}), \tag{2.16}$$

with known (A_i, B_i) for $i \in \{0, 1, \ldots, p\}$, where $\theta_{i,t}^{\text{tr}}$ denotes the i-th entry of the vector at t. We refine Θ_t from collected data. The associated algorithm is presented in Chapter 6.

- *For models of the form (M3), we consider $d(x)$ is L_d-Lipschitz with a known Lipschitz constant L_d, and its possible range $\mathcal{D}(x)$ is refined from collected data via information of the graph of the function $d(\cdot)$. The associated algorithm thus relies on a non-parametric representation of uncertainty $d(x)$, and is presented in Chapter 5.*

To tackle challenge (C2), it is common to restrict the search of optimal policy to a specific class of feedback policies. In this dissertation, we will focus on the following two design approaches used in MPC literature:

(P1) Design with affine state feedback policies, resulting in the so-called *tube* MPC algorithms [52, 15, 53, 46, 19], and

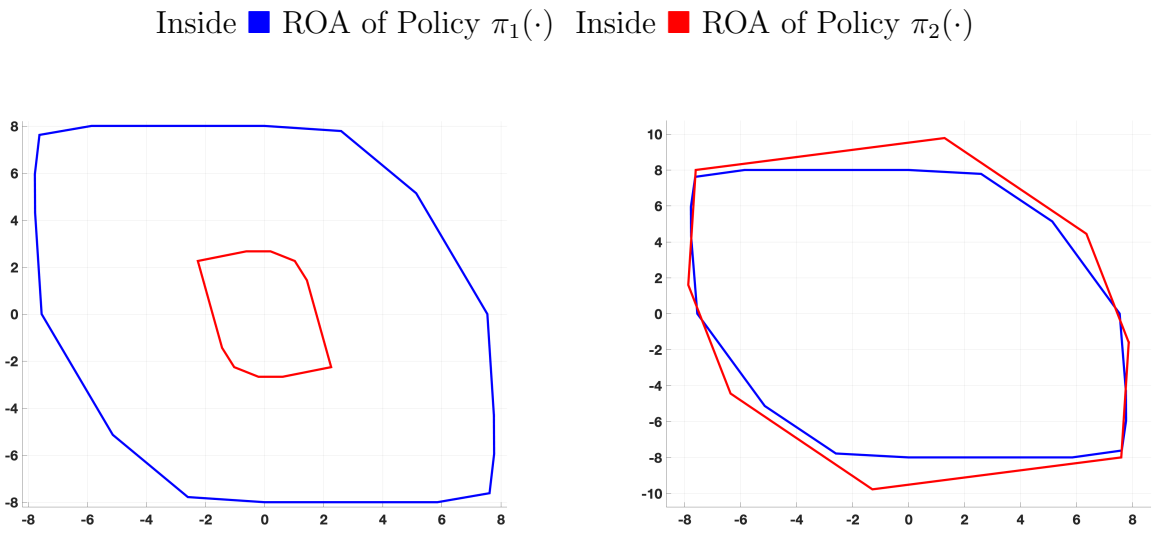
(P2) Design with affine disturbance feedback policies [54, 55, 32, 33].

And finally, to tackle challenge (C3), the terminal set $\mathcal{X}_N$ and the terminal cost function $Q(\cdot)$ are chosen appropriately to ensure feasibility and stability properties.

2.6 The Key, Often Misused Word: "Conservative"

Throughout this dissertation, we will be using the word conservative when discussing the properties of any control policy. When a control policy is deemed more conservative over another, it often implies that it has a smaller region of attraction (ROA) than the other. Although more formally defined in the subsequent chapters, roughly speaking, the ROA of a policy denotes the set of states from which the policy is able to safely complete the control task of stabilization, while satisfying the imposed state and input constraints.

Consider the two ROAs for a control problem obtained with policies $\pi_1(\cdot)$ and $\pi_2(\cdot)$, as shown in Fig. 2.1. In this dissertation, we adhere to the following two methods of comparison



(a) Policy $\pi_1(\cdot)$ is more conservative than policy $\pi_2(\cdot)$.

(b) Conservatism comparison is inconclusive.

Figure 2.1: Comparison of the regions of attraction obtained with policies $\pi_1(\cdot)$ and $\pi_2(\cdot)$.

between any two policies $\pi_1(\cdot)$ and $\pi_2(\cdot)$ while analyzing the two separate situations shown in Fig. 2.1:

- In a scenario such the one shown in Fig. 2.1a, we deem policy $\pi_1(\cdot)$ more conservative than policy $\pi_2(\cdot)$. This is because the ROA of $\pi_1(\cdot)$ is entirely contained in the ROA of $\pi_2(\cdot)$.
- In a scenario such the one shown in Fig. 2.1b, the ROA comparison is inconclusive for a comment on conservatism, as there are non intersecting parts of the ROAs.

Defining Conservatism via Cost

The notion of conservatism of a policy can also be defined in terms of the closed-loop cost of trajectories. For example, consider the case of Fig. 2.1a. The average closed-loop cost of

100 Monte-Carlo runs from 100 sampled initial conditions inside the ROA of $\pi_2(\cdot)$ can be obtained as shown in Fig. 2.2. We see from Fig. 2.2 that policy $\pi_2(\cdot)$ valid over a smaller

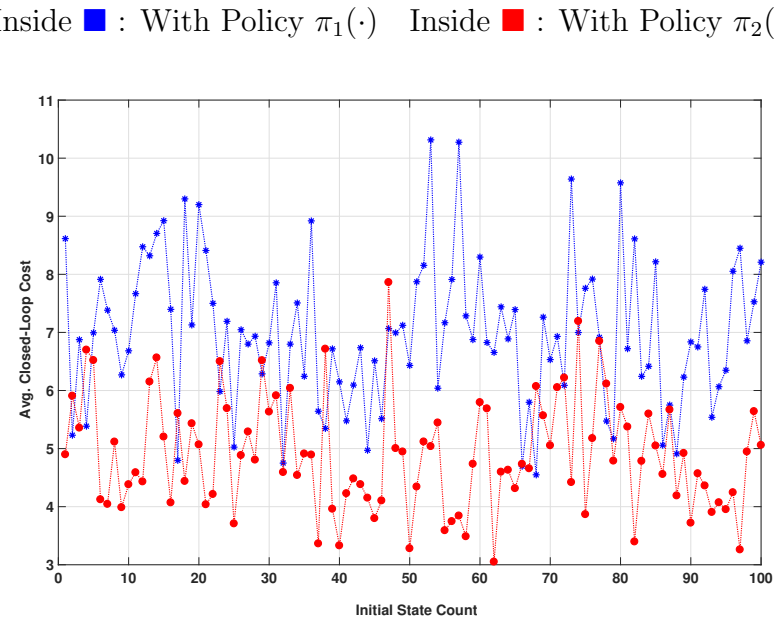


Figure 2.2: Comparison of the avg. closed-loop costs obtained with policies $\pi_1(\cdot)$ and $\pi_2(\cdot)$.

ROA attains a better average closed-loop cost over policy $\pi_1(\cdot)$ in about 95% of the cases[1]. If the notion of conservatism is defined in terms of the closed-loop cost, policy $\pi_1(\cdot)$ would be deemed more conservative over policy $\pi_2(\cdot)$. However, throughout this dissertation we will continue to define conservatism through the lens of the size of the ROA and not the cost of the closed-loop trajectories. We will instead use the cost of closed-loop trajectories to quantify *performance* of a control policy.

In the following sections we elaborate the MPC design details for each of the models (M1)-(M3) considered, with the policy parametrizations (P1)-(P2).

2.7 (M1)/(P1): Linear Time-Invariant Systems with Additive Disturbance, Design with Affine State Feedback Policies - Tube MPC

A classical robust MPC approach for linear time-invariant systems with additive disturbance, i.e., systems of the form (2.9), is the "tube" MPC approach. In tube MPC algorithms for

[1]Note that Fig. 2.2 has been generated only for explanatory purposes in this remark. The plots are *not* from closed-loop simulations of a chosen system inside ROAs shown in Fig. 2.1.

this class of systems, the system state x_t is split into a nominal and an error state as

$$x_t = \bar{x}_t + e_t, \tag{2.17}$$

and decoupled dynamics equations for each is used. The primary objective is to focus on the nominal state dynamics during MPC design, while containing all possible evolutions of the error state within a "tube" around the predicted nominal trajectory. The imposed constraints on the nominal states and inputs in the MPC optimization problem are appropriately shrunk from $\mathcal{X}$ and $\mathcal{U}$, ensuring that (2.7d)-(2.7e) are satisfied as long as the nominal constrains are satisfied. We specifically focus on two types of tube MPC algorithms, namely, shrinking tube MPC and rigid tube MPC. They are elaborated next. For a discussion on additional tube MPC algorithms such as homothetic and parametric tube MPC, see [56, 57, 58].

Remark 2.4 *The number of decision variables in both shrinking tube and rigid tube MPC algorithms scale linearly with the length of the prediction horizon, i.e., N.*

2.7.1 Shrinking Tube MPC

In shrinking tube MPC [52], the nominal state is set to the measured state of the system at any timestep t, i.e.,

$$\bar{x}_t = x_t,$$

and then the size of the imposed constraint sets on the predicted nominal states/inputs along the prediction horizon are progressively shrunk to guarantee the satisfaction of (2.7d)-(2.7e).

Policy Parametrization, Nominal and Error Dynamics

The input policy parametrization for shrinking tube MPC is given by:

$$u(x_t) = Kx_t + v_t, \tag{2.18}$$

with a fixed feedback gain K such that $(A + BK)$ is stable and v_t is an auxiliary control input which a decision variable. The dynamics of the nominal and error states from (2.17) using policy (2.18) are then obtained as:

$$\bar{x}_{t+1} = (A + BK)\bar{x}_t + Bv_t, \tag{2.19a}$$
$$e_{t+1} = (A + BK)e_t + w_t. \tag{2.19b}$$

Shrinking Tube MPC Problem

The MPC optimization problem solved focuses on imposing constraints on predicted nominal states and inputs obtained from (2.18) and (2.19a), such that accounting for all possible errors

from (2.19b) maintains the satisfaction of (2.7d)-(2.7e). This MPC problem is given by:

$$\min_{v_{t|t},\dots,v_{t+N-1|t}} \sum_{k=t}^{t+N-1} \ell\left(\bar{x}_{k|t}, v_{k|t}\right) + Q(\bar{x}_{t+N|t}) \tag{2.20a}$$

$$\text{s.t.,} \quad \bar{x}_{k+1|t} = (A+BK)\bar{x}_{k|t} + Bv_{k|t},\ k = \{t, t+1, \dots, t+N-1\}, \tag{2.20b}$$

$$\bar{x}_{k|t} \in \mathcal{X} \ominus \bigoplus_{i=0}^{k-t-1} (A+BK)^i \mathbb{W}, \tag{2.20c}$$

$$K\bar{x}_{k|t} + v_{k|t} \in \mathcal{U} \ominus \bigoplus_{i=0}^{k-t-1} K(A+BK)^i \mathbb{W}, \tag{2.20d}$$

$$\bar{x}_{t+N|t} \in \mathcal{X}_N \ominus \bigoplus_{i=0}^{N-1} (A+BK)^i \mathbb{W}, \tag{2.20e}$$

$$\forall k = \{t+1, \dots, t+N-1\}, \tag{2.20f}$$

$$\bar{x}_{t|t} = x_t, \tag{2.20g}$$

and the MPC controller

$$u^\star_{t|t}(x_t) = Kx_t + v^\star_{t|t}$$

is applied to system (2.9) in closed-loop. A robust positive invariant terminal set $\mathcal{X}_N$ with a terminal policy $u = Kx$, and a terminal cost function $Q(\cdot)$ which is a Lyapunov function for the closed-loop system $x^+ = (A+BK)x$ addresses challenge (C3). Note that in (2.20c)-(2.20d) the size of the imposed constraint sets on the nominal states and inputs shrink as k increases along the horizon. This is the shrinking tube property of this algorithm. Constraints (2.20c)-(2.20d) ensure the satisfaction of (2.7d) by system (2.9).

2.7.2 Rigid Tube MPC

Recall the nominal and error state decomposition from (2.17). In rigid tube MPC [53], the size of the imposed constraint sets on the predicted nominal states/inputs along the prediction horizon are kept fixed, while guaranteeing the satisfaction of (2.7d)-(2.7e). The nominal state at any timestep t is a decision variable in the MPC problem.

Policy Parametrization, Nominal and Error Dynamics

The input policy parametrization for rigid tube MPC is given by:

$$u(x_t) = K\bar{x}_t + K_e e_t + v_t, \tag{2.21}$$

with fixed feedback gains K and K_e such that $(A+BK)$ and $(A+BK_e)$ are stable and v_t is an auxiliary control input which a decision variable. The dynamics of the nominal and error

states from (2.17) using policy (2.21) are:

$$\bar{x}_{t+1} = (A + BK)\bar{x}_t + Bv_t, \tag{2.22a}$$
$$e_{t+1} = (A + BK_e)e_t + w_t. \tag{2.22b}$$

Rigid Tube MPC Problem

Consider a robust positive invariant set for the error dynamics (2.22b), denoted by $\mathcal{E}$. Given a measured state x_t, the choice of the nominal state $\bar{x}_t = \bar{x}_{t|t}$ is to be made such that the error state at timestep t, i.e., $e_t = e_{t|t}$ satisfies

$$e_{t|t} \in \mathcal{E},$$

thus restricting the evolution of the error state in $\mathcal{E}$ at all future times. The MPC optimization problem is formulated by imposing constraints on predicted nominal states and inputs given by the following:

$$\bar{x}_{k|t} \in \mathcal{X} \ominus \mathcal{E}, \tag{2.23a}$$
$$K\bar{x}_{k|t} + v_{k|t} \in \mathcal{U} \ominus K_e\mathcal{E}, \tag{2.23b}$$

obtained from (2.21) and (2.22a). One can demonstrate that if (2.23) are satisfied, (2.7d)-(2.7e) must be satisfied. The resulting robust MPC problem is:

$$\min_{\bar{x}_{t|t}, v_{t|t}, \ldots, v_{t+N-1|t}} \sum_{k=t}^{t+N-1} \ell\left(\bar{x}_{k|t}, v_{k|t}\right) + Q(\bar{x}_{t+N|t}) \tag{2.24a}$$
$$\text{s.t.,} \quad \bar{x}_{k+1|t} = (A + BK)\bar{x}_{k|t} + Bv_{k|t}, \tag{2.24b}$$
$$\bar{x}_{k|t} \in \mathcal{X} \ominus \mathcal{E}, \tag{2.24c}$$
$$K\bar{x}_{k|t} + v_{k|t} \in \mathcal{U} \ominus K_e\mathcal{E}, \tag{2.24d}$$
$$\bar{x}_{t+N|t} \in \bar{\mathcal{X}}_N \subset \mathcal{X} \ominus \mathcal{E}, \tag{2.24e}$$
$$\forall k = \{t, \ldots, t+N-1\}, \tag{2.24f}$$
$$e_{t|t} \in \mathcal{E}, \tag{2.24g}$$

where $\bar{\mathcal{X}}_N$ is the terminal constraint set for the nominal state. The MPC controller

$$u^\star_{t|t}(x_t) = K\bar{x}^\star_{t|t} + K_e e_{t|t} + v^\star_{t|t}.$$

is then applied to system (2.9) in closed-loop. A control invariant terminal set $\bar{\mathcal{X}}_N$ and a terminal cost function $Q(\cdot)$ which is a control Lyapunov function in the terminal set addresses challenge (C3). Note that in (2.24c)-(2.24d) the size of the imposed constraint sets on the nominal states and inputs remain fixed as k increases along the horizon. This is the rigid tube property of this algorithm. Constraints (2.24c)-(2.24d) ensure the satisfaction of (2.7d) by system (2.9).

Example 2.1 (Shrinking Tube vs Rigid Tube) *We now present an example that compares shrinking and rigid tube MPC algorithms in terms of their ROA. The corresponding ROAs for this problem are approximated by gridding the state-space in a* 50×50 *grid of initial conditions and taking the convex hull of the initial conditions for which* (2.20) *and* (2.24) *are feasible. Consider finding shrinking and rigid tube MPC solutions to the robust infinite horizon optimal control problem given by*

$$\begin{aligned} \min_{u_0, u_1(\cdot), \ldots} \quad & \sum_{t \geq 0} 10 \left\| \bar{x}_t \right\|_2^2 + 2 \left\| u_t(\bar{x}_t) \right\|_2^2 \\ \text{s.t.}, \quad & x_{t+1} = Ax_t + Bu_t(x_t) + w_t, \\ & \bar{x}_{t+1} = A\bar{x}_t + Bu_t(\bar{x}_t), \\ & \begin{bmatrix} -8 \\ -8 \\ -4 \end{bmatrix} \leq \begin{bmatrix} x_t \\ u_t(x_t) \end{bmatrix} \leq \begin{bmatrix} 8 \\ 8 \\ 4 \end{bmatrix}, \\ & \forall w_t \in \mathbb{W}, \ t = 0, 1, 2, \ldots, \end{aligned} \tag{2.25}$$

with the disturbance support set $\mathbb{W} = \{w : \|w\|_\infty \leq 0.1\}$*, where*

$$A = \begin{bmatrix} 1 & 0.05 \\ 0 & 1 \end{bmatrix}, \ B = \begin{bmatrix} 0 \\ 1.1 \end{bmatrix}.$$

We pick $N = 5$*. The matrix* K *in* (2.18) *and* (2.21) *is chosen as the infinite horizon LQR gain with* $Q = \text{diag}(10, 10)$ *and* $R = 2$*. Matrix* $K_e = K$ *in* (2.21)*. The terminal set* $\bar{\mathcal{X}}_N$ *in* (2.24) *is chosen as the maximal positive invariant set for the nominal dynamics* (2.22a) *under policy* $u = K\bar{x}_t$*, and the set* $\mathcal{E}$ *in* (2.24) *is chosen as the minimal robust positive invariant set for the error dynamics* (2.22b)*. In Fig. 2.3 we show the comparison of the approximate ROAs.*

Fig. 2.3 suggests that near the edge of the constraint sets, the constant (and larger) tightening of the state constraints in (2.24) *via* $\mathcal{E}$ *results in infeasibility of* (2.24)*, where a slowly increased tightening along the prediction horizon results in a feasible controller obtained as a solution to* (2.20) *from the corresponding states. This indicates a lower conservatism of the shrinking tube MPC for the considered example, as seen from its larger ROA in Fig. 2.3, which entirely contains the ROA of the rigid tube MPC. However, recall that the terminal set in rigid tube MPC can also be a control invariant set, which can enlarge its ROA if chosen appropriately*[2]*. Moreover, computing a (nominal) control invariant* $\bar{\mathcal{X}}_N$ *in* (2.24) *with a fixed linear feedback policy is cheaper than computing a robust positive invariant* $\mathcal{X}_N$ *in* (2.20)*. Thus, the use of shrinking tube vs rigid tube is dependent on the specific problem at hand, computational constraints to the designer, etc.*

Remark 2.5 *The rigid tube MPC's property of robust control synthesis based on the containment of error in an invariant set around the nominal trajectory is similar to methods*

[2]An example of such a set can be found in [59], which iteratively enlarges a control invariant terminal set constructed with collected trajectory data and provides feasibility and stability guarantees of the controller.

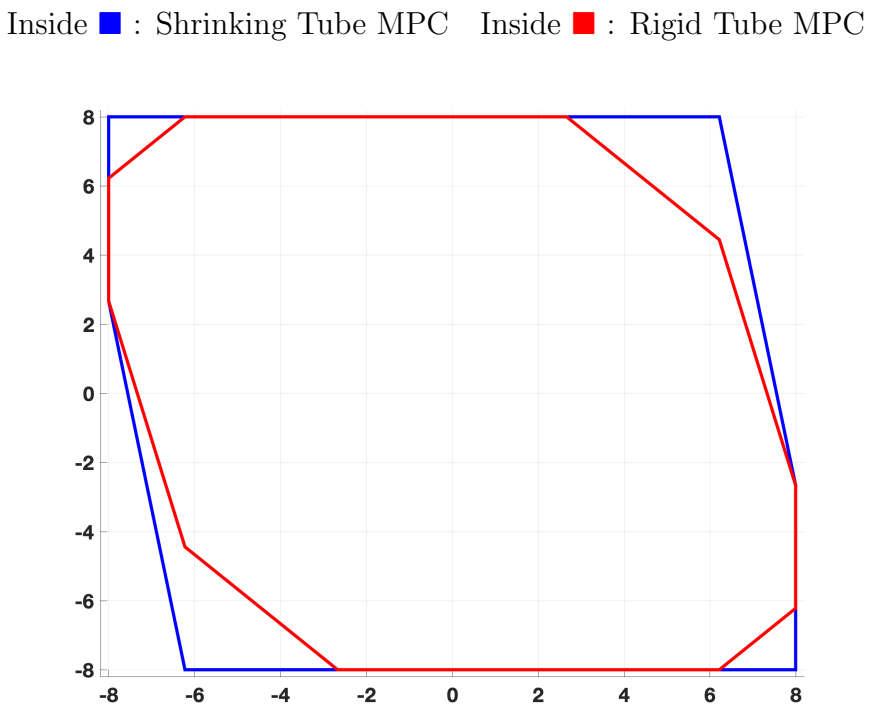


Figure 2.3: Comparison of the approx. ROAs of shrinking tube and rigid tube MPC. The shrinking tube MPC obtains reduced conservatism in this case over the rigid tube MPC.

developed later, such as [60, 61, 62, 63], which are primarily for nonlinear systems. These methods exploit planner-tracker hierarchies during control design and maintain the tracking error in an invariant set around a planner generated trajectory.

2.8 (M1)/(P2): Linear Time-Invariant Systems with Additive Disturbance, Design with Affine Disturbance Feedback Policies

The tube MPC algorithms in Section 2.7 utilize a fixed feedback gain along the prediction horizon in (2.18) and (2.21). Instead, utilizing an input policy of the form

$$u_t(x_t) = K_t x_t + v_t, \tag{2.26}$$

with time varying feedback gains K_t and optimizing over these gains along the prediction horizon can lower the conservatism of (2.20) and (2.24). However, optimizing over these feedback gains directly in a state feedback based robust MPC synthesis results in a non-convex problem, as shown in [55, Proposition 3]. A convex synthesis is enabled using the affine disturbance feedback policy parametrization [54, 55], which is proven to be equivalent

to (2.26) for models (M1) [55, Section 5]. We describe this parametrization and the associated robust MPC algorithm next.

Policy Parametrization

For all $k \in \{t, \dots, t+N-1\}$ over the MPC horizon of length N, the control policy parametrization in this case is given as:

$$u_{k|t}(x_{k|t}) = \sum_{j=t}^{k-1} M_{k,j|t} w_{j|t} + v_{k|t}, \tag{2.27}$$

where $M_{k|t}$ are the *planned* feedback gains at timestep t and $v_{k|t}$ are the auxiliary inputs, both of which are decision variables. Let us define the stacked disturbances along the MPC prediction horizon as

$$\mathbf{w}_t = \begin{bmatrix} w_{t|t}^\top & w_{t+1|t}^\top & \cdots & w_{t+N-1|t}^\top \end{bmatrix}^\top \in \mathbb{R}^{nN}.$$

Then the sequence of predicted inputs from (2.27) can be stacked together as:

$$\mathbf{u}_t = \mathbf{M}_t \mathbf{w}_t + \mathbf{v}_t$$

at any timestep t, where matrices $\mathbf{M}_t \in \mathbb{R}^{mN \times nN}$ and $\mathbf{v}_t \in \mathbb{R}^{mN}$ are obtained by arranging the decision variables as follows:

$$\mathbf{M}_t = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ M_{t+1,t} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ M_{t+N-1,t} & \cdots & M_{t+N-1,t+N-2} & 0 \end{bmatrix}, \quad \mathbf{v}_t = \begin{bmatrix} v_{t|t} \\ v_{t+1|t} \\ \vdots \\ v_{t+N-1|t} \end{bmatrix}. \tag{2.28}$$

MPC Problem

Using policy (2.27), problem (2.7) in this case is simplified to solving:

$$\begin{aligned} \min_{\mathbf{M}_t, \mathbf{v}_t} \quad & \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}, v_{k|t}) + Q(\bar{x}_{t+N|t}) \\ \text{s.t.,} \quad & x_{k+1|t} = Ax_{k|t} + Bu_{k|t} + w_{k|t}, \\ & \bar{x}_{k+1|t} = A\bar{x}_{k|t} + Bv_{k|t}, \\ & u_{k|t}(x_{k|t}) = \sum_{j=t}^{k-1} M_{k,j|t} w_{j|t} + v_{k|t}, \\ & x_{k|t} \in \mathcal{X},\ u_{k|t}(x_{k|t}) \in \mathcal{U}, \\ & x_{t+N|t} \in \mathcal{X}_N, \\ & \forall w_{k|t} \in \mathbb{W}, \\ & \forall k = \{t, \dots, t+N-1\}, \\ & x_{t|t} = \bar{x}_t = x_t. \end{aligned} \tag{2.29}$$

A robust positive invariant terminal set $\mathcal{X}_N$ with the terminal policy $u = Kx$, and a terminal cost function $Q(\cdot)$ which is a Lyapunov function for the closed-loop system $x^+ = (A+BK)x$ addresses challenge (C3). Problem (2.29) scales quadratically [64] in the number of decision variables with the horizon N. An efficient way to solve (2.29) using duality of convex programs [65, 66, 67] is shown in [55]. We present an example of such a reformulation next.

Example 2.2 *We show an example of how to convert (2.29) into a tractable convex optimization problem using duality of convex programs. The example is from [6, Example 15.3]. Consider the system*

$$x_{t+1} = x_t + u_t + w_t, \tag{2.30}$$

and let

$$\begin{aligned}\mathcal{U} &= \{u : -1 \leq u \leq 1\},\\ \mathbb{W} &= \{w : -1 \leq w \leq 1\}.\end{aligned}$$

The objective of the player is to play three input moves so that $x_3 \in \mathcal{X}_N$, with the set $\mathcal{X}_N$ given by

$$\mathcal{X}_N = \{x : -1 \leq x \leq 1\}.$$

The terminal constraint can be rewritten as

$$\begin{aligned}&x_3 = x_0 + u_0 + u_1 + u_2 + w_0 + w_1 + w_2 \in [-1, 1],\\ &\forall w_0 \in [-1, 1],\ \forall w_1 \in [-1, 1],\ \forall w_2 \in [-1, 1].\end{aligned}$$

The control inputs are parametrized in the past disturbances following (2.27) as

$$\begin{aligned}u_0 &= v_0,\\ u_1 &= v_1 + M_{1,0}w_0,\\ u_2 &= v_2 + M_{2,0}w_0 + M_{2,1}w_1.\end{aligned}$$

The input constraints and the terminal constraint are then rewritten as

$$x_0 + v_0 + v_1 + v_2 + (1 + M_{1,0} + M_{2,0})w_0 + (1 + M_{2,1})w_1 + w_2 \in [-1, 1], \tag{2.31a}$$
$$u_0 = v_0 \in [-1, 1], \tag{2.31b}$$
$$v_1 + M_{1,0}w_0 \in [-1, 1], \tag{2.31c}$$
$$v_2 + M_{2,0}w_0 + M_{2,1}w_1 \in [-1, 1], \tag{2.31d}$$
$$\forall w_0 \in [-1, 1],\ \forall w_1 \in [-1, 1],\ \forall w_2 \in [-1, 1]. \tag{2.31e}$$

Consider (2.31a) and the one sided inequality given by

$$\begin{aligned}&x_0 + v_0 + v_1 + v_2 + (1 + M_{1,0} + M_{2,0})w_0 + (1 + M_{2,1})w_1 + w_2 \leq 1,\\ &\forall w_0 \in [-1, 1],\ \forall w_1 \in [-1, 1],\ \forall w_2 \in [-1, 1].\end{aligned} \tag{2.32}$$

We replace (2.32) *with the most stringent constraint given by*

$$x_0 + v_0 + v_1 + v_2 + J^\star(M_{1,0}, M_{2,0}, M_{2,1}) \leq 1, \tag{2.33}$$

where we have used

$$\begin{aligned} J^\star(M_{1,0}, M_{2,0}, M_{2,1}) = \max_{w_0,w_1,w_2} \quad & (1 + M_{1,0} + M_{2,0})w_0 + (1 + M_{2,1})w_1 + w_2 \\ \text{s.t.,} \quad & w_0 \in [-1, 1],\ w_1 \in [-1, 1],\ w_2 \in [-1, 1]. \end{aligned} \tag{2.34}$$

For fixed $M_{1,0}, M_{2,0}, M_{2,1}$, (2.34) *is a linear program and can be replaced in* (2.32) *by its dual*

$$x_0 + v_0 + v_1 + v_2 + d^\star(M_{1,0}, M_{2,0}, M_{2,1}) \leq 1,$$

where

$$\begin{aligned} d^\star(M_{1,0}, M_{2,0}, M_{2,1}) = \min_{\lambda_i^u, \lambda_i^l} \quad & \lambda_0^u + \lambda_1^u + \lambda_2^u + \lambda_0^l + \lambda_1^l + \lambda_2^l \\ \text{s.t.,} \quad & 1 + M_{1,0} + M_{2,0} + \lambda_0^l - \lambda_0^u = 0, \\ & 1 + M_{2,1} + \lambda_1^l - \lambda_1^u = 0, \\ & 1 + \lambda_2^l - \lambda_2^u = 0, \\ & \lambda_i^u \geq 0,\ \lambda_i^l \geq 0,\ i = 0, 1, 2, \end{aligned}$$

where λ_i^u *and* λ_i^l *are the dual variables corresponding to the upper and lower bounds of* w_i*, respectively. From strong duality [66, Chapter 5], [45, 68, 69, 70], we then impose the following equivalent constraints to satisfy* (2.33)-(2.34)

$$\begin{aligned} & x_0 + v_0 + v_1 + v_2 + \lambda_0^u + \lambda_1^u + \lambda_2^u + \lambda_0^l + \lambda_1^l + \lambda_2^l \leq 1, \\ & 1 + M_{1,0} + M_{2,0} + \lambda_0^l - \lambda_0^u = 0, \\ & 1 + M_{2,1} + \lambda_1^l - \lambda_1^u = 0, \\ & 1 + \lambda_2^l - \lambda_2^u = 0, \\ & \lambda_i^u \geq 0,\ \lambda_i^l \geq 0,\ i = 0, 1, 2. \end{aligned} \tag{2.35}$$

Thus, (2.35) *is a convex reformulation of* (2.32) *with finite number of constraints. Repeating this same procedure for the other side of the inequality, we see* (2.31a) *is reformulated as*

$$\begin{aligned} & x_0 + v_0 + v_1 + v_2 + \lambda_0^u + \lambda_1^u + \lambda_2^u + \lambda_0^l + \lambda_1^l + \lambda_2^l \leq 1, \\ & -x_0 - v_0 - v_1 - v_2 + \mu_0^u + \mu_1^u + \mu_2^u + \mu_0^l + \mu_1^l + \mu_2^l \leq 1, \\ & 1 + M_{1,0} + M_{2,0} + \lambda_0^l - \lambda_0^u = 0, \\ & 1 + M_{2,1} + \lambda_1^l - \lambda_1^u = 0, \\ & 1 + \lambda_2^l - \lambda_2^u = 0, \\ & -1 - M_{1,0} - M_{2,0} + \mu_0^l - \mu_0^u = 0, \\ & -1 - M_{2,1} + \mu_1^l - \mu_1^u = 0, \\ & -1 - \mu_2^l - \mu_2^u = 0, \\ & \lambda_i^u, \lambda_i^l, \mu_i^u, \mu_i^l \geq 0,\ i = 0, 1, 2, \end{aligned} \tag{2.36}$$

and (2.31c) *is reformulated as*

$$\begin{aligned} & v_1 + \nu_0^u + \nu_0^l \leq 1, \\ & -v_1 + \kappa_0^u + \kappa_0^l \leq 1, \\ & M_{1,0} + \kappa_1^l - \kappa_1^u = 0, \\ & \kappa_0^u, \kappa_0^l, \nu_0^u, \nu_0^l \geq 0, \end{aligned} \tag{2.37}$$

and (2.31d) *is reformulated as*

$$\begin{aligned} & v_2 + \rho_0^u + \rho_1^u + \rho_0^l + \rho_1^l \leq 1, \\ & -v_2 + \pi_0^u + \pi_1^u + \pi_0^l + \pi_1^l \leq 1, \\ & M_{2,0} + \rho_0^l - \rho_0^u = 0, \\ & M_{2,1} + \rho_1^l - \rho_1^u = 0, \\ & -M_{2,0} + \pi_0^l - \pi_0^u = 0, \\ & -M_{2,1} + \pi_1^l - \pi_1^u = 0, \\ & \rho_i^u, \rho_i^l, \pi_i^u, \pi_i^l \geq 0, \ \ i = 0, 1, \end{aligned} \tag{2.38}$$

and (2.31b) *remains unchanged. Solving* (2.35)-(2.38) *provides a solution to our problem.*

Example 2.3 *In order to show the reduction in conservatism with the policy parametrization* (2.27) *over* (2.18) *and* (2.21) *we again consider MPC solutions to the problem* (2.3)*, and compare the ROA of an MPC controller synthesized by solving* (2.29) *to the ROAs of shrinking tube and rigid tube MPCs, shown in Fig. 2.3. Parameters* Q, R, K *are the same as the ones in Example 2.1. The comparison is shown in Fig. 2.4. Fig. 2.4 demonstrates that the MPC policy obtained with the affine disturbance feedback parametrization attains the largest ROA, i.e., the lowest conservatism compared to both shrinking and rigid tube MPC.*

As a matter of fact, the disturbance feedback policy class (2.27) subsumes the policy classes (2.18) and (2.21), as shown in [55]. This explains its reduced conservatism over tube MPC approaches in Section 2.7, as seen in Fig. 2.4. Therefore, we use the disturbance feedback based robust MPC algorithm from Section 2.8 for robust adaptive MPC synthesis in Chapter 5 and Chapter 6, and for robust MPC design in conjunction with disturbance distribution support and environment constraint learning in Chapter 7 and Chapter 8, respectively.

2.9 (M2)/(P1): Linear Parameter Varying Systems, Design with Affine State Feedback Policies

Consider the state decomposition

$$x_t = \bar{x}_t + e_t$$

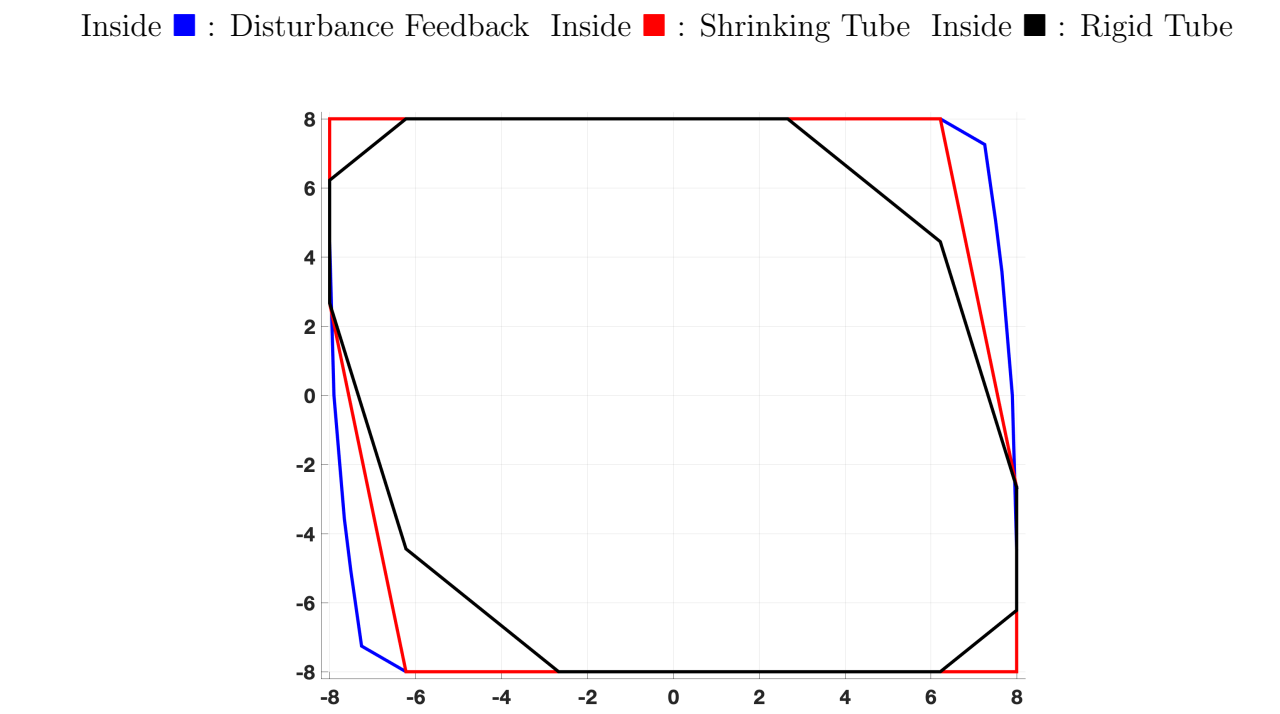


Figure 2.4: Comparison of the ROAs of shrinking tube MPC, rigid tube MPC and an MPC designed with affine disturbance parametrization (2.27).

from (2.17) for system (2.10), written as:

$$\bar{x}_{t+1} + e_{t+1} = (\bar{A} + \Delta_A)(\bar{x}_t + e_t) + (\bar{B} + \Delta_B)u_t + w_t, \tag{2.39}$$

where $\bar{x}_t$ is the certainty equivalent state. Clearly, if one defined the nominal model as (2.19a) or (2.22a), then we see from simple substitution that using the MPC policies from Section 2.7 in (2.39), a decomposition of the system dynamics into nominal (i.e., certainty equivalent) and error states' dynamics, as done in (2.19) and (2.22) cannot be carried out. In fact, it is not possible to cancel out the the product between the matrix uncertainty term Δ_A and the nominal state $\bar{x}_t$. Therefore, algorithms from Section 2.7 and Section 2.8 which rely on a decoupled set of nominal and error state dynamics cannot be utilized. An alternative set of algorithms are summarized next.

2.9.1 Ellipsoidal ROA Synthesis

Ellipsoidal ROA synthesis methods for system 2.10 are presented in works such as [71, 72, 73, 74]. These methods constrain the evolution of the system state inside an ellipsoidal robust positive invariant set under a chosen affine control policy, for all possible realizations of the uncertainty. The associated control gain in the MPC policy is computed typically by solving

linear matrix inequalities (LMI) [44], which ensures convex controller synthesis. A detailed summary of these algorithms can be found in [75, Section 5.2].

Policy Parametrization and Invariant Ellipsoid

Consider the following input policy parametrization:

$$u(x_t) = Kx_t + v_t,$$

where the feedback matrix K and the auxiliary input v_t are decision variables. In the invariant ellipsoidal approach, the decision variables K and v_t are chosen such that

$$x_{k|t} \in \mathcal{E} \subseteq \mathcal{X},\ \forall \Delta_A \in \mathcal{P}_A,\ \forall \Delta_B \in \mathcal{P}_B,\ \forall k \geq t+1, \tag{2.40}$$

if $x_t \in \mathcal{E}$, and $u_{k|t} \in \mathcal{U}$ for all $k \geq t$. Equation 2.40 implies that the set $\mathcal{E}$ is thus an ellipsoidal invariant set for the predicted states of system (2.10). Furthermore, thus restricting the system state's evolution to an invariant set allows for the possibility of solving the MPC optimization problem over an infinite horizon, as opposed to a finite horizon of N in (2.7). This ensures that $\mathcal{E}$ is an invariant set for the closed-loop states of the system as well. In order to better elaborate the MPC design approach, we focus on a specific example of the ellipsoidal ROA synthesis algorithm in [71]. This is presented next.

An Example Algorithm From [71]

The MPC proposed in [71] for system (2.10) considers $\mathbb{W} = \emptyset$, $v_t = 0$, and an additional simplifying assumption on the matrix uncertainties given by:

$$\begin{bmatrix} \Delta_A^{\text{tr}} & \Delta_B^{\text{tr}} \end{bmatrix} = \sum_{i=1}^{n_{ab}} \lambda_i \begin{bmatrix} \Delta_A^{(i)} & \Delta_B^{(i)} \end{bmatrix},\ \text{with } \lambda_i \geq 0,\ \sum_{i=1}^{n_{ab}} \lambda_i = 1. \tag{2.41}$$

The approach in [71] finds a solution to an infinite horizon robust optimal control problem for MPC synthesis. This infinite horizon problem is still solved at every timestep t, as the MPC policy improves in closed-loop due to the restriction of the search of input policies to the class of affine state feedback. This MPC optimization problem at any timestep t can is:

$$\begin{aligned} \min_{K} \max_{\Delta_A \in \mathcal{P}_A, \Delta_B \in \mathcal{P}_B} \quad & \sum_{k=t}^{\infty} x_{k|t}^\top Q_s x_{k|t} + x_{k|t}^\top K^\top R K x_{k|t} \\ \text{s.t.,} \quad & x_{k+1|t} = (A + BK) x_{k|t}, \\ & x_{k|t} \in \mathcal{X},\ Kx_{k|t} \in \mathcal{U}, \\ & \forall k = t, t+1, \ldots, \end{aligned} \tag{2.42}$$

with weight matrices $Q_s, R \succ 0$, and the state and input constraints in (2.42) considered as:

$$\mathcal{X} = \{x : \|x\|_2 \leq x_{\max}\}, \tag{2.43a}$$

$$\mathcal{U} = \{u : \|u\|_2 \leq u_{\max}\}, \tag{2.43b}$$

where $x_{\max}$ and $u_{\max}$ are known. Note that (2.42) minimizes a worst-case cost, as opposed to the nominal cost in (2.7). It is shown in [71, Section 3.2] that the solution K to (2.42) can be obtained as $K = YQ^{-1}$, with decision matrices $Y \in \mathbb{R}^{m\times n}$ and $Q \succ 0$ obtained as the solutions to the problem:

$$\min_{\gamma,Y,Q\succ 0} \quad \gamma$$

$$\text{s.t.,} \quad \begin{bmatrix} Q & Q(\bar{A}+\Delta_A^{(i)})^\top + Y^\top(\bar{B}+\Delta_B^{(i)})^\top & QQ_s^{1/2} & Y^\top R^{1/2} \\ (\bar{A}+\Delta_A^{(i)})Q + (\bar{B}+\Delta_B^{(i)})Y & Q & 0 & 0 \\ Q_s^{1/2} & 0 & \gamma I_n & 0 \\ R^{1/2}Y & 0 & 0 & \gamma I_m \end{bmatrix} \succeq 0, \tag{2.44a}$$

$$\begin{bmatrix} Q & ((\bar{A}+\Delta_A^{(i)})Q + (\bar{B}+\Delta_B^{(i)})Y)^\top \\ (\bar{A}+\Delta_A^{(i)})Q + (\bar{B}+\Delta_B^{(i)})Y & x_{\max}^2 I_n \end{bmatrix} \succeq 0, \tag{2.44b}$$

$$\begin{bmatrix} u_{\max}^2 I_m & Y \\ Y^\top & Q \end{bmatrix} \succeq 0, \tag{2.44c}$$

$$\begin{bmatrix} 1 & x_{t|t}^\top \\ x_{t|t} & Q \end{bmatrix} \succeq 0, \tag{2.44d}$$

$$\forall i = 1, 2, \ldots, n_{ab}.$$

Note that (2.44a) ensures the worst-case infinite horizon cost in (2.42) is bounded from above by γ, (2.44b) ensures the satisfaction of the state constraints (2.43a), (2.44c) ensures the satisfaction of the input constraints (2.43b), and (2.44d) ensures that the current state must be inside the invariant ellipsoid, as required in (2.40). The MPC policy is given by:

$$u_t^{\text{MPC}}(x_t) = Y^\star Q^{-1,\star} x_t, \tag{2.45}$$

where decision variables $Y^\star, Q^\star$ are function of the current state $x_t = x_{t|t}$. Under policy (2.45), the the invariant ellipsoid in (2.40) obtained as: [71, Eq. 31]

$$\mathcal{E}(x_t) = \{z : z^\top \gamma^\star Q^{-1,\star} z \leq \gamma^\star\}. \tag{2.46}$$

As (2.44) is solved at each timestep t, the invariant ellipsoid changes as a function of x_t. However each ellipsoid $\mathcal{E}(x_t)$ is invariant for (2.10) if the MPC policy (2.45) is rolled out for all future times without re-solving (2.44), as (2.44) provides a solution valid over an infinite prediction horizon.

Although computationally efficient (number of decision variables is not a function of the prediction horizon N, and typically scale polynomially with the dimension of the state-space [75, Chapter 5]), these algorithms often result in conservative controller behavior arising from the design limitation into the space of ellipses, e.g., in (2.43). Polytopic and homothetic tube MPC methods with affine or piecewise affine state feedback policy parametrizations are introduced in [15, 76, 16, 19] to address such conservatism inherent to ellipsoidal ROA based methods. We elaborate these algorithms next.

2.9.2 Polytopic Tube MPC

Although many forms of polytopic tube MPC algorithms exist in literature, we focus on the ones in [75, Section 5.6], [16, 19]. These have a similar approach when compared to other polytopic approaches such as the algorithms in [17, 18, 76]. We omit explicitly showing the dimensions of a set of matrices and vectors in the following sections for the purpose of notational simplicity. See [75, 19] for these details.

Policy Parametrization

An input policy parametrization for this class of algorithms used in [75, Section 5.6], [19] is given by:

$$u(x_t) = Kx_t + Lv_t + c_t, \text{ with additional dynamics } v_{t+1} = \sum_{i=1}^{n_{ab}} \lambda_i (M_i v_t + S_i w_t), \tag{2.47}$$

with a stabilizing feedback gain K for system (2.10), where L, M_i, S_i, c_t and v_0 are decision variables and λ_is satisfy (2.41). Matrices L, M_i and S_i for all $i = \{1, 2, \ldots, n_{ab}\}$ are chosen offline to maximize the volume of the terminal set [75, Eq. 5.111]. The terminal set $\mathcal{X}_N$ in these algorithms is typically chosen as an ellipsoidal invariant set, which is constructed by solving an LMI, based on methods similar to [74, 77]. The use of these additional degrees of freedom in (2.47) via the choice of L, M_is and S_is is thus for the purpose of enlarging the size of the ROA of the robust MPC [75, Section 5.6].

Designing Polytopic Reachable Sets

Polytopic and homothetic tube MPC methods make use of the following sets:

$$\mathcal{X}_{k|t} = \{x : Vx \leq \alpha_{k|t}\}, \ k = t+1, t+2, \ldots, t+N, \tag{2.48}$$

with $\mathcal{X}_{t|t} = x_t$, where V is a matrix chosen offline, and vectors $\alpha_{k|t}$ are the decision variables in the online MPC synthesis problem. See [75, Chapter 5] for details on how to choose V offline. Define the 1-step robust reachable set from $\mathcal{X}_{k|t}$ under the MPC policy (2.47) for all possible matrices (A, B), and disturbances lying in a known support $\mathbb{W}$ as follows:

$$\begin{aligned}\mathcal{R}_{k+1|t}(v_{k|t}, c_{k|t}) = \{x_{k+1|t} : \exists x_{k|t} \in \mathcal{X}_{k|t}, \exists w_{k|t} \in \mathbb{W}, \exists \Delta_A \in \mathcal{P}_A, \exists \Delta_B \in \mathcal{P}_B, \\ \text{s.t., } x_{k+1|t} = (\bar{A} + \Delta_A)x_{k|t} + (\bar{B} + \Delta_B)(Kx_{k|t} + Lv_{k|t} + c_{k|t}) + w_{k|t}\},\end{aligned}$$

for all $k = t, t+1, \ldots, t+N-1$. The online decision variables $v_{t|t}, c_{k|t}$, and $\alpha_{k|t}$ defining the sets in (2.48) are then chosen such that

$$\begin{aligned}&\mathcal{X}_{k+1|t} \supseteq \mathcal{R}_{k+1|t}(v_{k|t}, c_{k|t}), \\ &\forall k = t, t+1, \ldots, t+N-1,\end{aligned} \tag{2.49}$$

thus constraining all possible evolutions of the predicted states within the design sets (2.48). Constraints (2.49) are imposed by transforming them into the following constraints (obtained using Farka's lemma):

$$\begin{aligned} &H_i V = V f_1(\bar{A}, \bar{B}, \mathcal{P}_A, \mathcal{P}_B, M_i, S_i), \\ &\alpha_{k+1|t} \geq H_i \alpha_{k|t} + f_2(w, V, \mathcal{P}_A, \mathcal{P}_B, c_{k|t}), \\ &H_i \geq 0, \\ &\forall k \in \{t+1, \ldots, t+N-1\}, \\ &\forall i \in \{1, 2, \ldots, n_{ab}\}, \end{aligned} \tag{2.50}$$

where the exact form of the functions f_1 and f_2 vary depending on the specific algorithm used. See [19, Theorem 8], [75, Eq. 5.112] for details on these variations. Matrices H_i for all $i \in \{1, 2, \ldots, n_{ab}\}$ have non negative entries, and these are decision variables chosen offline. See [75, Section 5.6] for further details on these offline optimization problems.

Imposing the State and Input Constraints

The online decision variables $v_{k|t}, c_{k|t}, \alpha_{k|t}$ are also constrained by the requirement of satisfying the state and input constraints in (2.7d). Expressing these state and input constraints in (2.7d) alternatively as:

$$Cx_t + Du_t \leq b, \tag{2.51}$$

for matrices C, D, b of appropriate dimensions, constraints (2.51) are satisfied robustly along the prediction horizon by imposing the feasibility of a set of constraints that can be summarized by the following:

$$H_c \alpha_{k|t} + D c_{k|t} \leq b, \tag{2.52a}$$
$$\forall k = \{t, t+1, \ldots, t+N-1\},$$
$$H_c \geq 0, \alpha_{t|t} \geq V \begin{bmatrix} x_t \\ v_{t|t} \end{bmatrix}, \tag{2.52b}$$
$$H_c V = \begin{bmatrix} C + DK & DL \end{bmatrix}, \tag{2.52c}$$
$$H_i \alpha_{t+N|t} + V f_3(w_t) \leq \alpha_{t+N|t}, \tag{2.52d}$$
$$\forall i = \{1, 2, \ldots, n_{ab}\},$$
$$H_c \alpha_{t+N|t} \leq 1_h, \tag{2.52e}$$

where (2.52e) are the terminal constraints in the MPC problem, i.e., (2.7e), and $f_3(\cdot)$ is linear in w_t. The non-negative entries of matrix H_c with h number of rows are decision variables chosen offline. See [75, Section 5.6] for further details on the offline problem solved.

Remark 2.6 *We have based (2.52) on [75, Section 5.6]. The corresponding equations in [16, 19] are slightly different as they design matrix V in (2.48) to shape the cross sections of the reachable sets for error state $e_t = x_t - \bar{x}_t$.*

Polytopic Tube MPC Problem

The resulting polytopic tube MPC problem solved can be expressed as:

$$\begin{aligned}\min_{v_{t|t},\boldsymbol{\alpha}_t,\mathbf{c}_t} \quad \max_{\substack{\Delta_A\in\mathcal{P}_A,\Delta_B\in\mathcal{P}_B \\ w_{t|t},w_{t+1|t},\ldots,w_{t+N-1|t}}} \quad & \mathcal{L}(x_{k|t}|_{k=t}^{t+N}, u_{k|t}(x_{k|t})|_{k=t}^{t+N-1}) \\ \text{s.t.,} \quad & x_{k+1|t} = (A+BK)x_{k|t} + B(Lv_{k|t}+c_{k|t}),\ k=t,\ldots,t+N-1, \\ & (2.50)-(2.52)\end{aligned} \tag{2.53}$$

where $\boldsymbol{\alpha}_t = \{\alpha_{t|t},\ldots,\alpha_{t+N|t}\}$, $\mathbf{c}_t = \{c_{t|t},\ldots,c_{t+N-1|t}\}$, and the cost function $\mathcal{L}$ is obtained via solving an LMI such that it is an upper bound on the worst-case infinite horizon cost. See [75, Lemma 5.8] for additional details. Note that (2.53) minimizes a worst-case cost. A nominal cost as used in (2.7) can also be minimized alternatively.

Although convex control synthesis is allowed by these algorithms, the number of constraints in the MPC problem from (2.50) and (2.52) increase noticeably with the horizon length, as shown in [75, Table 5.2]. The number of constraints can be lowered with simpler choices of V, and thus limiting the shape of the tube cross sections in (2.48). This however induces additional conservatism in the design. Thus, managing the computational complexity vs conservatism trade-off is a key challenge in the design of these classes of polytopic tube MPC algorithms.

Example 2.4 (Polytopic Tube MPC vs Ellipsoidal ROA Synthesis) *Throughout the relevant discussion in this dissertation until Chapter 4, we will consider finding MPC solutions to the robust infinite horizon optimal control problem given by*

$$\begin{aligned}\min_{u_0,u_1(\cdot),\ldots} \quad & \sum_{t\geq 0} 10\,\|\bar{x}_t\|_2^2 + 2\,\|u_t(\bar{x}_t)\|_2^2 \\ \text{s.t.,} \quad & x_{t+1} = Ax_t + Bu_t(x_t) + w_t, \text{ with } A = \bar{A}+\Delta_A,\ B = \bar{B}+\Delta_B, \\ & \bar{x}_{t+1} = \bar{A}\bar{x}_t + \bar{B}u_t(\bar{x}_t), \\ & \begin{bmatrix}-8\\-8\\-4\end{bmatrix} \leq \begin{bmatrix}x_t\\u_t(x_t)\end{bmatrix} \leq \begin{bmatrix}8\\8\\4\end{bmatrix}, \\ & \forall w_t\in\mathbb{W},\ \forall\Delta_A\in\mathcal{P}_A,\ \forall\Delta_B\in\mathcal{P}_B, \\ & t=0,1,2,\ldots,\end{aligned} \tag{2.54}$$

with disturbance set $\mathbb{W} = \{w : \|w\|_\infty \leq 0.1\}$, *where*

$$\bar{A} = \begin{bmatrix}1 & 0.15\\0.1 & 1\end{bmatrix},\ \bar{B} = \begin{bmatrix}0.1\\1.1\end{bmatrix},\ A = \begin{bmatrix}1 & 0.05\\0 & 1\end{bmatrix},\ B = \begin{bmatrix}0\\1.1\end{bmatrix}.$$

For solving (2.54), *we consider the matrix uncertainty sets given by*

$$\mathcal{P}_A : \text{conv}\Big(\begin{bmatrix}0 & \pm 0.1\\ \pm 0.1 & 0\end{bmatrix}\Big),\ \text{(4 matrices)} \quad \mathcal{P}_B : \text{conv}\Big(\begin{bmatrix}0\\ \pm 0.1\end{bmatrix}, \begin{bmatrix}\pm 0.1\\0\end{bmatrix}\Big).\ \text{(4 matrices)}$$

For ellipsoidal ROA synthesis method [71], the disturbance set $\mathbb{W} = \emptyset$. *The comparison of* $\mathcal{E}(x_t)$ *from* (2.46) *evaluated at*[3] $x_t = [5, 4]^\top$ *and the terminal set of [19] is shown in Fig. 2.5. We see from Fig. 2.5 that the terminal set of [19] is about 2x in volume compared to the*

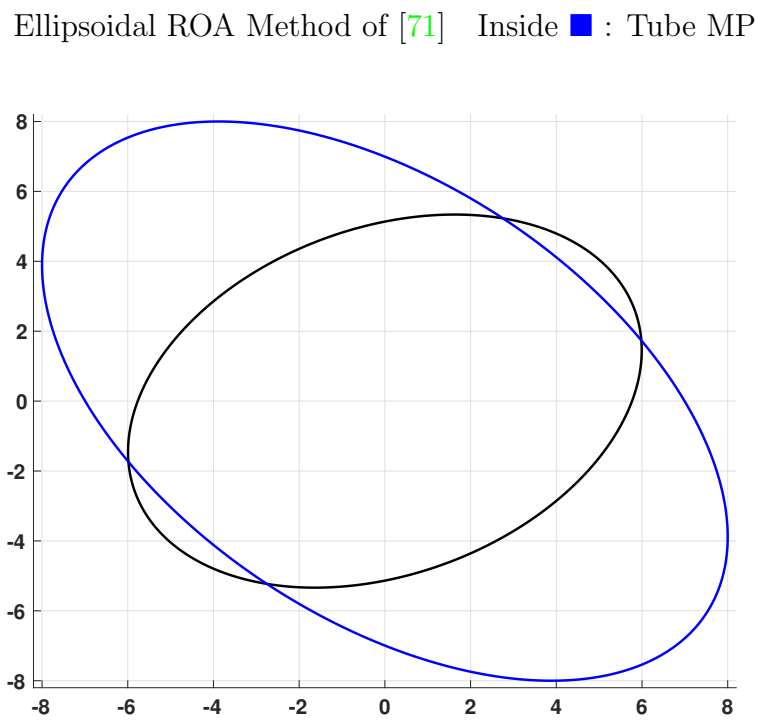


Figure 2.5: Comparison of the ellipsoidal ROA of [71] and the ROA of [19]. The ellipsoidal terminal set is the ROA of [19] for this example, as pointed out in Remark 2.7.

ROA of [71], despite the presence of the additional additive disturbance considered in [19].

Remark 2.7 *Note, the matrix* V *in* (2.48) *required for computing optimized tubes could not be found by following [19, Remark 10], even under an upper bound of 100 on its number of rows. See [19, Section 3], [75, Lemma 5.7] for a detailed explanation to this fact from the viewpoint of spectral radius of the resulting dynamics. This yielded the full MPC strategy in [19, Section 4.2], which solves an optimization problem similar to* (2.53)*, impractical for obtaining an online MPC solution to the above problem. Therefore, a larger ROA than the terminal set could not be obtained for the considered example. This highlights the computational drawbacks of polytopic tube MPC methods, such as [16, 19].*

[3] This is one of the largest ROAs found after sampling 100 states x_t and computing (2.46).

2.10 (M2)/(P2): Linear Parameter Varying Systems, Design with Affine Disturbance Feedback Policies

In this section we present an overview of robust MPC design for models of the form (M2) with policy parametrization (P2). First, we provide a brief introduction to robust MPC design via system level synthesis, which obtains an alternative parametrization of the system response via the use of policy. Subsequently, we introduce two novel algorithms presented in Chapter 3 and Chapter 4 of this dissertation.

2.10.1 System Level Synthesis

An alternative set of approaches for robust MPC design for system 2.10 based on system level synthesis (SLS) [78, 79] are presented in [26, 80, 81, 82]. Unlike typical MPC algorithms which optimize over inputs and states as decision variables, this method obtains robust constraint satisfaction by optimizing over closed-loop system responses. Consider an alternative representation of the matrix uncertainty in (2.10), given by:

$$\|\Delta_A\|_\infty \leq \epsilon_A, \ \|\Delta_B\|_\infty \leq \epsilon_B.$$

The used policy parametrization in these methods is given by:

$$\mathbf{u}_t = \mathbf{M}_t \mathbf{w}_t, \tag{2.55}$$

with the associated definitions in (2.28). Briefly speaking, the aforementioned SLS based robust MPC approaches such as [26, 80, 82] solve an MPC problem that can be summarized roughly to be of the following form:

$$\begin{aligned}
\min_{\mathbf{\Phi}_t^x, \mathbf{M}_t, \tau, \gamma, \beta} \quad & \mathcal{L}(\mathbf{\Phi}_t^x, \mathbf{M}_t, x_t) && \text{(2.56a)}\\
\text{s.t.,} \quad & \begin{bmatrix} I_n - Z\bar{A} & -Z\bar{B} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_t^x \\ \mathbf{M}_t \end{bmatrix} = I_n, && \text{(2.56b)}\\
& f_1(\mathbf{\Phi}_t^x, \mathbf{M}_t, \mathcal{X}, \mathcal{U}, \mathcal{X}_N, \mathbb{W}, \tau, \gamma, \beta) \leq 0, && \text{(2.56c)}\\
& f_2(\|f_3(\mathbf{\Phi}_t^x, \mathbf{M}_t, \mathcal{X}, \mathcal{U}, \mathcal{X}_N, \epsilon_A, \epsilon_B, x_t)\|) \leq f_4(\gamma, \tau), && \text{(2.56d)}
\end{aligned}$$

where Z is the block downshift operator. We can summarize (2.56) as follows: Cost function (2.56a) is similar to the nominal cost minimized in (2.7a). Alternatively, worst-case and expected costs may be chosen. Dynamics (2.56b) is the subspace equation, which indicates that $\mathbf{\Phi}_t^x$ is the predicted response of system (2.10) under the policy (2.55) and ignoring the effect of model mismatch. Constraints (2.56c)-(2.56d) jointly ensure the satisfaction of constraints $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{X}_N$ robustly by the predicted states and inputs for all possible realizations of the model mismatch and disturbances. These are equivalent to (2.7d) and

(2.7e). Functions $f_i(\cdot)$ for $i = 1, 2, \ldots, 4$ depend on the exact control design formulation, which ensure that satisfying (2.56) results in a convex optimization problem for online control synthesis. The parameters β, τ, γ are chosen offline before control design, typically using grid search methods [27, 26].

Although online control synthesis is computationally efficient [80, 26], these algorithms can turn out to be conservative, as shown in [33, 32]. This conservatism primarily stems from obtaining sufficient conditions in (2.56c)-(2.56d) required for satisfying state and input constraints (2.7d) and (2.7e). As suggested by the results in the recent papers [26, 80], polytopic tube MPC algorithms described in Section 2.9.2, such as [15, 76, 19], etc., can obtain larger ROAs compared to SLS based robust MPC methods. We therefore focus on robust MPC algorithm design taking polytopic tube MPC methods as benchmarks, and do not focus on designing SLS based algorithms in the subsequent chapters of this dissertation.

2.10.2 Novel Algorithms Proposed in Chapter 3-4

To better exploit the computational complexity vs conservatism trade-off, in this dissertation we present two novel algorithms for robust MPC design for system (2.10). In both these algorithms we use the affine disturbance feedback policy.

Net-Additive Uncertainty Representation and Shrinking Horizons

In the first algorithm, proposed in Chapter 3, we exploit the concept of net-additive uncertainty representation by expressing system 2.10 as:

$$x_{t+1} = \bar{A}x_t + \bar{B}u_t + \tilde{w}_t,$$

with the net-additive uncertainty term given by

$$\tilde{w}_t = (\Delta_A x_t + \Delta_B u_t + w_t).$$

We bound the quantity $\tilde{w}_t$ with its worst-case bound over the state and input spaces as follows:

$$\begin{aligned}
\max_{t\geq 0} \|\tilde{w}_t\| &\leq \max_{t\geq 0, \Delta_A\in\mathcal{P}_A, \Delta_B\in\mathcal{P}_B} (\|\Delta_A x_t\| + \|\Delta_B u_t\| + \|w_t\|), && \text{(2.57a)}\\
&\leq \max_{t\geq 0, \Delta_A\in\mathcal{P}_A, \Delta_B\in\mathcal{P}_B} (\|\Delta_A\|_p\|x_t\| + \|\Delta_B\|_p\|u_t\| + \|w_t\|), && \text{(2.57b)}\\
&= \max_{\Delta_A\in\mathcal{P}_A, \Delta_B\in\mathcal{P}_B} (\|\Delta_A\|_p\|x\|_{\max} + \|\Delta_B\|_p\|u\|_{\max} + \|w\|_{\max}), && \text{(2.57c)}\\
&= \tilde{w}_{\max},
\end{aligned}$$

where in (2.57a) we have used the triangle inequality and in (2.57b) the consistency property of induced norms. Values of $\|x\|_{\max}$, $\|u\|_{\max}$ and $\|w\|_{\max}$ in (2.57c) can be obtained from compact constraints (3.1e) and $\mathbb{W}$. For example, as we use polytopic sets $\mathcal{X}$,$\mathcal{U}$ and $\mathbb{W}$ in

Chapter 3, we obtain these maximum values by evaluating the norms at the vertices of the corresponding polytopes.

The two key design features of the proposed algorithm in Chapter 3 and their benefits can be summarized as follows: First, we use the conservative bound (2.57) along the prediction horizon in formulating the robust MPC problem following Section 2.8. This allows for fast MPC synthesis without the increase in the number of imposed constraints with N, as is the case with methods described in Section 2.9.2. And second, for the design of the terminal set $\mathcal{X}_N$, we do not over-approximate the system uncertainty. This lowers the conservatism of the terminal policy over methods described in Section 2.9.2, which use an LMI synthesized ellipsoidal terminal set. Our terminal set computation may be expensive, but it is computed offline. Note that due the discrepancy in the uncertainty representations used along the prediction horizon and in the terminal set, shrinking horizons are used to ensure the feasibility of the MPC problem. See Chapter 3 for details. Due to the lack of such a shrinking horizon strategy, SLS based methods such as [26] lose recursive feasibility upon using a similar terminal set. This is alleviated with the use of shrinking horizons in the most recent SLS based work [80, 82].

Optimization-Based Constraint Tightening

Our second algorithm is presented in Chapter 4. This algorithm circumvents the conservative worst-case bounding of a net additive uncertainty term $\tilde{w}_t$ and uses constraint tightening strategy which is a function of predicted nominal states and inputs in the MPC optimization problem, i.e.,

$$\bar{x}_{k|t} \in \mathcal{X} \ominus \mathcal{E}^{\text{tight}}_{k|t}(\bar{x}_{i|t}|_{i=t}^{k}, u_{i|t}(\bar{x}_{i|t})|_{i=t}^{k-1}, \boldsymbol{\Delta}, \mathbb{W}),\ k = t+1, t+2, \ldots, t+N, \tag{2.58}$$

where $\boldsymbol{\Delta}$ denotes a contribution to the constraint tightening from the uncertainty in system matrices. The two key design benefits are: First, tightening strategy (2.58) lowers conservatism over using the worst-case bound from (2.57) uniformly across the state-space, as it depends on optimization variables in the MPC problem, i.e., $(\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t}))$. See Chapter 4 for details of this novel constraint tightening formulation. And second, the online control synthesis problem is computationally efficient over methods in Section 2.9.2, as we show in Table 4.2. Computing the bounds $\boldsymbol{\Delta}$ required in (2.58) may be computationally expensive, but these are computed offline before online control synthesis.

Both of the algorithms in Chapter 3-4 solve convex optimization problems for online controller synthesis. The details of terminal set selection to address challenge (C3) is in Section 3.5. With detailed numerical simulations, we demonstrate in Chapter 3 and Chapter 4 that our proposed algorithms can obtain an improved computational complexity vs conservatism trade-off, over methods elaborated in Section 2.9.2 and Section 2.10.1. The presented numerical comparisons are with [15, 19, 27]. We use the two novel algorithms proposed in Chapter 3-4 for robust adaptive MPC synthesis when there is uncertainty present in the system dynamics matrices A and B. We present an example of the above in Chapter 6.

2.11 (M3)/(P1) or (P2): Linear Time-Invariant Systems with State Dependent Additive Uncertainty

We jointly discuss the cases of using (P1) or (P2) in this scenario. The reformulation of problem (2.7) in this scenario is given by:

$$\begin{aligned}
\min_{u_{t|t},u_{t+1|t}(\cdot),\ldots,u_{t+N-1|t}(\cdot)} \quad & \textstyle\sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})) + Q(\bar{x}_{t+N|t}) \\
\text{s.t.,} \quad & x_{k+1|t} = Ax_{k|t} + Bu_{k|t}(x_{k|t}) + d(x_{k|t}), \\
& \bar{x}_{k+1|t} = A\bar{x}_{k|t} + Bu_{k|t}(\bar{x}_{k|t}) + \bar{d}(\bar{x}_{k|t}), \\
& x_{k+1|t} \in \mathcal{X}, \\
& u_{k|t}(x_{k|t}) \in \mathcal{U}, \\
& \forall d(x_{k|t}) \in \mathcal{D}(x_{k|t}), \\
& \forall k = \{t, t+1, \ldots t+N-1\}, \\
& x_{t+N|t} \in \mathcal{X}_N, \\
& x_{t|t} = \bar{x}_{t|t} = x_t,
\end{aligned} \tag{2.59}$$

where $\mathcal{D}(x_t)$ is a state dependent compact set where the uncertainty $d(x_t)$ is guaranteed to lie, and $\bar{d}(\bar{x}_t)$ denotes the certainty equivalent (nominal) estimate of uncertainty at any point $\bar{x}_t$ along the nominal trajectory. Given the sets $\mathcal{D}(x_{k|t})$ for all $k \in \{t, t+1, \ldots, t+N\}$, standard robust MPC approaches such as the ones in Section 2.7 and Section 2.8 can be used for control synthesis. Data-driven methods such as GP regression have been used in literature to learn and refine these sets $\mathcal{D}(x_{k|t})$ [7, 9, 8]. However, they lack closed-loop guarantees of robust constraint satisfaction.

In Chapter 5 of this dissertation we present an adaptive MPC algorithm which utilizes Lipschitz nature of the function $d(\cdot)$, and then designs a robust MPC based on Section 2.8. Our proposed algorithm guarantees recursive feasibility of the MPC problem, while lowering its conservatism by successively refining the domain of the uncertainty.

Remark 2.8 *A Python based repository which implements a basic set of robust MPC algorithms, including the ones presented next in Chapter 3 and Chapter 4 is now publicly available at: https://github.com/monimoyb/RMPCPy. The results from the examples in this chapter can be obtained by running the scripts in this repository.*

Chapter 3

A Simple Robust MPC for Linear Parameter Varying Systems

This chapter is based on the published work [33]. In this chapter we propose a simple algorithm for robust MPC design for linear parameter varying (LPV) systems subject to a bounded additive disturbance. The algorithm uses a worst-case uncertainty bound of the system for constraint tightening along the prediction horizon, which is computed by lumping up the contribution of matrix uncertainties and the additive disturbances into one "net-additive term". The worst-case value of this quantity is computed over the sets of feasible states and inputs, as shown in (2.57).

3.1 Summary of Contributions

We show that the naive net-additive uncertainty approach can lead to an efficient MPC design, if the terminal constraints are appropriately chosen and an adaptive horizon strategy is adopted. Our method can also be used to obtain a single roll-out policy for robust constraint satisfaction, without re-solving the MPC problem. The key contributions of this chapter are:

- We split the constraint tightening into two cases based on the horizon length. For a horizon length of one, the robust MPC problem is solved exactly without over-approximating the uncertainty in the system. For larger horizons, we lump the model uncertainty into a net-additive component and compute constraint tightenings along the prediction horizon based on its worst-case bound.
- We solve a set of tractable convex optimization problems online using an adaptive horizon approach for robust MPC synthesis. With an appropriately constructed terminal set and a terminal cost we prove recursive feasibility of the controller synthesis problem in closed-loop and input to state stability of the origin.

3.2 Problem Formulation

We consider the linear systems of the form (2.10). We are interested in synthesizing a robust MPC for (2.10), by repeatedly solving the following optimal control problem:

$$\min_{U_t(\cdot)} \quad \sum_{k=t}^{t+N-1} \ell\left(\bar{x}_{k|t}, u_{k|t}\left(\bar{x}_{k|t}\right)\right) + Q(\bar{x}_{t+N|t}) \tag{3.1a}$$

$$\text{s.t.,} \quad \bar{x}_{k+1|t} = \bar{A}\bar{x}_{k|t} + \bar{B}u_{k|t}(\bar{x}_{k|t}), \tag{3.1b}$$

$$x_{k+1|t} = Ax_{k|t} + Bu_{k|t}(x_{k|t}) + w_{k|t}, \tag{3.1c}$$

$$\text{with } A = \bar{A} + \Delta_A, B = \bar{B} + \Delta_B, \tag{3.1d}$$

$$H^x x_{k|t} \leq h^x, H^u u_{k|t}(x_{k|t}) \leq h^u, \tag{3.1e}$$

$$x_{t+N|t} \in \mathcal{X}_N, \tag{3.1f}$$

$$\forall w_{k|t} \in \mathbb{W},\ \forall \Delta_A \in \mathcal{P}_A,\ \forall \Delta_B \in \mathcal{P}_B, \tag{3.1g}$$

$$\forall k \in \{t, t+1, \ldots, (t+N-1)\},$$

$$x_{t|t} = \bar{x}_{t|t} = x_t,$$

with $U_t(\cdot) = \{u_{t|t}, u_{t+1|t}(\cdot), \ldots, u_{t+N-1|t}(\cdot)\}$, and applying the optimal MPC policy

$$u_t^{\text{MPC}}(x_t) = u_{t|t}^\star, \tag{3.2}$$

to system (2.10) in closed-loop, where $x_{k|t}$ is the predicted state at timestep k for any possible uncertainty realization, obtained by applying the set of predicted input policies $\{u_{t|t}, u_{t+1|t}(\cdot), \ldots, u_{k-1|t}(\cdot)\}$ to system (2.10), and $\{\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})\}$ denote the nominal state and corresponding input respectively. The constraints (3.1e)-(3.1f) are satisfied for all uncertainty realizations in (3.1g), where $H^x \in \mathbb{R}^{s\times n}, h^x \in \mathbb{R}^s, H^u \in \mathbb{R}^{o\times m}$ and $h^u \in \mathbb{R}^o$ parametrize *compact* sets. Finally, the stage cost $\ell(x,u) = x^\top Px + u^\top Ru$, and the terminal cost $Q(x) = x^\top P_N x$.

3.2.1 Control Policy Parametrization

Recall the quantity $\tilde{w}_t$ and its bounding from (2.57). For all predicted steps $k \in \{t, t+1, \ldots, t+N-1\}$ over the MPC horizon, the control policy is chosen as per (P2) in Chapter 2, i.e.,

$$u_{k|t}(x_{k|t}) = \sum_{l=t}^{k-1} M_{k,l|t}\tilde{w}_{l|t} + \bar{u}_{k|t}, \tag{3.3}$$

where $M_{k|t}$ are the feedback gains at timestep t and $\bar{u}_{k|t} = u_{k|t}(\bar{x}_{k|t})$ are the nominal inputs. Then the sequence of predicted inputs can be written as $\mathbf{u}_t = \mathbf{M}_t^{(N)}\tilde{\mathbf{w}}_t + \bar{\mathbf{u}}_t^{(N)}$, where $\mathbf{M}_t^{(N)} \in \mathbb{R}^{mN\times nN}$ and $\bar{\mathbf{u}}_t^{(N)} \in \mathbb{R}^{mN}$ are shown in (2.28), and

$$\mathbf{u}_t = [u_{t|t}^\top, u_{t+1|t}^\top(\cdot), \ldots, u_{t+N-1|t}^\top(\cdot)]^\top,$$

$$\tilde{\mathbf{w}}_t = \begin{bmatrix}\tilde{w}_{t|t}^\top & \tilde{w}_{t+1|t}^\top & \ldots \tilde{w}_{t+N-1|t}^\top\end{bmatrix}^\top,$$

with $\|\tilde{\mathbf{w}}_t\| \leq \tilde{\mathbf{w}}_{\max}$ for all $t \geq 0$, where $\tilde{\mathbf{w}}_{\max}$ is obtained from (2.57).

Approach Insight

We design a simple and computationally efficient shrinking tube MPC leveraging worst-case bounds of this net-additive uncertainty *only* along the prediction horizon, and an *exact* system uncertainty representation for the construction of the terminal set. Notice that robust MPC strategies such as [55, 52] using the net-additive uncertainty bounds *both* along the prediction horizon and for the computation of the terminal set, can be extremely conservative as pointed out in [83, 84]. Therefore, numerous alternative strategies such as polytopic, homothetic and elastic tube MPC [15, 16, 76] have been introduced to lower this conservatism by circumventing the net-additive uncertainty bounds, as we discussed in Chapter 2. This however increases the online computation times of these algorithms [75, 26], as we pointed out in Section 2.9.2. In Section 3.4, we show with numerical simulations that our approach balances this trade-off between conservatism and computational complexity. In the example under study, we obtain about 15x online computation speedup over the polytopic tube MPC method of [15], while stabilizing about 98% of its region of attraction. Additionally, our region of attraction is about 28% larger than that of the state-of-the-art polytopic tube MPC in [19], with about a 50% increase in online computation speeds.

3.2.2 Terminal Set Construction

We present the construction of the terminal set $\mathcal{X}_N$ in this section to address challenge (C3) mentioned in Chapter 2. Consider a linear state feedback policy for constructing $\mathcal{X}_N$

$$\kappa_N(x) = Kx, \tag{3.4}$$

where $K \in \mathbb{R}^{m \times n}$ is the feedback gain. Recall the sets $\mathcal{P}_A$ and $\mathcal{P}_B$ from (2.13). We define

$$\mathcal{P}_{A_\Delta} = \{A_m : A_m = \bar{A} + \Delta_A,\ \forall \Delta_A \in \mathcal{P}_A\},$$
$$\mathcal{P}_{B_\Delta} = \{B_m : B_m = \bar{B} + \Delta_B,\ \forall \Delta_B \in \mathcal{P}_B\}.$$

Under policy (3.4), the closed-loop system dynamics matrix considered for constructing the terminal set satisfies

$$A^{\mathrm{cl}} = A + BK \in \mathcal{P}_{A_\Delta} \oplus K\mathcal{P}_{B_\Delta}.$$

Assumption 3.1 *$A_m^{\mathrm{cl}} = (A_m + B_m K)$ is stable for all $A_m \in \mathcal{P}_{A_\Delta}$ and $B_m \in \mathcal{P}_{B_\Delta}$.*

Using Assumption 3.1, the terminal set $\mathcal{X}_N$ can then be computed as the maximal robust positive invariant set for

$$x_{t+1} = (A_m + B_m K)x_t + w_t,$$

for all $A_m \in \mathcal{P}_{A_\Delta}, B_m \in \mathcal{P}_{B_\Delta}$, and for all $w_t \in \mathbb{W}$. That is for all $x \in \mathcal{X}_N$ we have that

$$\begin{aligned} &H^x x \leq h^x,\ H^u K x \leq h^u \text{ and } (A_m + B_m K)x + w \in \mathcal{X}_N, \\ &\forall A_m \in \mathcal{P}_{A_\Delta},\ \forall B_m \in \mathcal{P}_{B_\Delta},\ \forall w \in \mathbb{W}. \end{aligned} \tag{3.5}$$

Fixed point iteration algorithms to numerically compute (3.5) can be found in [6, 75]. Note that these sets of algorithms typically have no convergence guarantees [85].

3.2.3 MPC Problem with Adaptive Horizon

We now present the MPC reformulation of (3.1) which guarantees recursive feasibility and Input to State Stability. Note, the terminal set $\mathcal{X}_N$ is robustly invariant to all uncertainty of the form: $\forall \Delta_A \in \mathcal{P}_A,\ \forall \Delta_B \in \mathcal{P}_B,\ \forall w \in \mathbb{W},\ \forall t \geq 0$, when the state feedback policy $\kappa_N(x) = Kx$ is used in closed-loop with system (2.10). However, along the prediction horizon we synthesize bound (2.57) using more conservative tightenings from Hölder's and triangle inequalities, and the induced norm consistency property. Thus the uncertainty bounds along the horizon over-approximate the effect of the true uncertainty used to compute the terminal set. This implies that the classical shifting argument [6, Chapter 12] for recursive MPC feasibility cannot be used in this case. To resolve this issue, we solve a set of N convex optimization problems at any t for control synthesis, with the prediction horizon $N_t \in \{1, 2, \ldots, N\}$. If one of these N problems is feasible at timestep 0, we guarantee feasibility of at least one of them for all $t \geq 0$.

We first use policy (3.3) to reformulate the robust state constraints in (3.1) along and at the end of the prediction horizon. Let the terminal set $\mathcal{X}_N$ in (3.5) be defined by $\mathcal{X}_N = \{x : H_N^x x \leq h_N^x\}$, with $H_N^x \in \mathbb{R}^{r \times n}, h_N^x \in \mathbb{R}^r$. For a horizon length of N_t, we denote matrices $\mathbf{F}^x = \text{diag}(I_{N_t-1} \otimes H^x, H_N^x) \in \mathbb{R}^{(s(N_t-1)+r) \times nN_t}$ and $\mathbf{f}^x = [(h^x)^\top, (h^x)^\top, \ldots, (h_N^x)^\top]^\top \in \mathbb{R}^{s(N_t-1)+r}$. Also denote the set $\tilde{\mathbf{W}} = \{\tilde{\mathbf{w}} \in \mathbb{R}^{nN_t} : \|\tilde{\mathbf{w}}_t\| \leq \tilde{\mathbf{w}}_{\max}\}$. Then we consider the following two cases as[1]:

Case 1: $N_t = 1$**:**

$$\max_{\substack{w_t \in \mathbb{W} \\ \Delta_A \in \mathcal{P}_A \\ \Delta_B \in \mathcal{P}_B}} H_N^x(\bar{A} + \Delta_A)x_t + (\bar{B} + \Delta_B)\bar{\mathbf{u}}_t^{(1)} + w_t) \leq h_N^x, \tag{3.6a}$$

Case 2: $N_t \geq 2$**:**

$$\max_{\tilde{\mathbf{w}}_t \in \tilde{\mathbf{W}}} \mathbf{F}^x \left(\bar{\mathbf{A}} x_t + \mathbf{C}\bar{\mathbf{u}}_t^{(N_t)} + (\mathbf{C}\mathbf{M}_t^{(N_t)} + \mathbf{G})\tilde{\mathbf{w}}_t \right) \leq \mathbf{f}^x, \tag{3.6b}$$

where matrices $\bar{\mathbf{A}}, \mathbf{C}$ and $\mathbf{G}$ are defined in the Appendix.

Remark 3.1 *In (3.6a) we exactly propagate the system uncertainty for robustification. This ensures the feasibility of (3.6a) inside $\mathcal{X}_N$, which is a robust positive invariant set computed*

[1]Note that the dimensions of $\mathbf{F}^x$, $\mathbf{f}^x$, $\bar{\mathbf{A}}$, $\mathbf{C}$, $\mathbf{G}$ and $\tilde{\mathbf{w}}_t$ vary depending on N_t. We omit showing this dependence explicitly for brevity.

from (3.5) *also using the exact uncertainty representation. As such uncertainty propagation is computationally intense over multi step predictions, in* (3.6b) *we over-approximate system uncertainty using bounds* (2.57).

Now, denote the matrices $\mathbf{H}^u = I_{N_t} \otimes H^u \in \mathbb{R}^{oN_t \times mN_t}$, and $\mathbf{h}^u = [(h^u)^\top, (h^u)^\top, \dots, (h^u)^\top]^\top \in \mathbb{R}^{oN_t}$. Once the state constraints are formulated, the input constraints in (3.1) along the prediction horizon can be written as:

$$\max_{\tilde{\mathbf{w}}_t \in \tilde{\mathbf{W}}} \mathbf{H}^u \left(\mathbf{M}_t^{(N_t)} \tilde{\mathbf{w}}_t + \bar{\mathbf{u}}_t^{(N_t)} \right) \leq \mathbf{h}^u, \tag{3.7}$$

for $N_t \in \{1, 2, \dots, N\}$. Using (3.6)-(3.7), we solve at any t:

$$\begin{aligned} & V_{t \to t+N_t}^{\mathrm{MPC}}(x_t, N_t) := \\ & \min_{\mathbf{M}_t^{(N_t)}, \bar{\mathbf{u}}_t^{(N_t)}} \quad \begin{bmatrix} (\bar{\mathbf{x}}_t^{(N_t)})^\top & (\bar{\mathbf{u}}_t^{(N_t)})^\top \end{bmatrix} \bar{Q}^{(N_t)} \begin{bmatrix} \bar{\mathbf{x}}_t^{(N_t)} \\ \bar{\mathbf{u}}_t^{(N_t)} \end{bmatrix} \\ & \text{s.t.,} \qquad \bar{\mathbf{x}}_t^{(N_t)} = \bar{\mathbf{A}} x_t + \mathbf{C} \bar{\mathbf{u}}_t^{(N_t)}, \\ & \qquad\quad (3.6a), (3.7) \text{ if } N_t = 1, \text{ else } (3.6b), (3.7), \\ & \qquad\quad \forall k = \{t, t+1, \dots, t+N_t-1\}, \\ & \qquad\quad \bar{x}_{t|t} = x_t, \end{aligned} \tag{3.8}$$

for $N_t \in \{1, 2, \dots, N\}$, where $\bar{Q}^{(N_t)} = \mathrm{diag}(I_{N_t} \otimes P, P_N, I_{N_t} \otimes R)$. We reformulate (3.8) as a convex program with standard duality arguments. After solving (3.8) for $N_t \in \{1, 2, \dots, N\}$, we set

$$N_t^\star = \arg \min_{\bar{N} \in \{1,2,\dots,N\}} V_{t \to t+N_t}^{\mathrm{MPC}}(x_t, \bar{N}). \tag{3.9}$$

Afterwards, we pick the solution associated with $N_t^\star$, and apply the corresponding optimal input

$$u_{t|t}^\star(x_t) = u_t^\star(x_t) = \bar{u}_{t|t}^\star, \tag{3.10}$$

to system (2.10), with $V_{t \to t+N_t^\star}^{\mathrm{MPC}}(x_t, N_t^\star) = J^\star(x_t)$. We then resolve (3.8) at $(t+1)$ for $N_{t+1} \in \{1, 2, \dots, N\}$.

3.3 Feasibility and Stability

In this section we prove the feasibility and stability properties of the proposed robust MPC.

3.3.1 Feasibility

Theorem 3.1 *Consider the closed-loop system* (2.10) *and* (3.10). *Let problem* (3.8) *be feasible at timestep $t = 0$ for some horizon length $N_t \in \{1, 2, \dots, N\}$. Then problem* (3.8) *is feasible at all timesteps $t \geq 1$ for some horizon length $N_t \in \{1, 2, \dots, N\}$, possibly time-varying.*

Proof *Assume that at timestep t problem (3.8) is feasible, and let $N_t^\star$ be the optimal horizon. We then consider the following two cases:*
Case 1: *($N_t^\star = 1$) Consider the robust state constraints (3.6a):*

$$\max_{\substack{w_t \in \mathbb{W} \\ \Delta_A \in \mathcal{P}_A, \Delta_B \in \mathcal{P}_B}} H_N^x((\bar{A} + \Delta_A)x_t + (\bar{B} + \Delta_B)\bar{\mathbf{u}}_t^{(1)} + w_t) \leq h_N^x. \tag{3.11}$$

We find h_N^x where the max is attained by using duality. Let us denote the corresponding optimal input policy by

$$u_{t|t}^\star(x_t) = \bar{u}_{t|t}^\star. \tag{3.12}$$

Now, let policy (3.12) be applied to (2.10) in closed-loop, so that the system reaches the terminal set $\mathcal{X}_N$. Consider solving (3.11) at this step with a horizon length of $N_{t+1} = 1$. As (3.6b) uses the same representation of the uncertainty as done in Section 3.2.2, a candidate policy at timestep $(t+1)$ is

$$u_{t+1|t+1}(x_{t+1}) = Kx_{t+1}, \tag{3.13}$$

which is a feasible solution to (3.8) under constraint (3.11).
Case 2: *($N_t^\star \geq 2$) Let us denote the sequence of optimal input policies from timestep t as $\{u_{t|t}^\star, u_{t+1|t}^\star(\cdot), \cdots, u_{t+N_t^\star-1|t}^\star(\cdot)\}$. Consider a candidate policy sequence at the next time instant:*

$$U_{t+1}(\cdot) = \{u_{t+1|t}^\star(\cdot), \ldots, u_{t+N_t^\star-1|t}^\star(\cdot)\}. \tag{3.14}$$

Now using standard MPC shifting arguments [52, 15, 55], sequence (3.14) is a feasible policy sequence at timestep $(t+1)$ for problem (3.8), with horizon length $N_{t+1} = N_t^\star - 1$.

3.3.2 Stability

To prove the stability of the origin in closed-loop, we first introduce the following set of assumptions and definitions.

Assumption 3.2 *Denote the set of state and input constraints in (3.1e) as $\mathcal{X}$ and $\mathcal{U}$, respectively. We assume that the convex, compact sets $\mathcal{X}$,$\mathcal{U}$ and $\mathbb{W}$ contain the origin in their interior.*

Recall the notion of N-Step Robust Controllable Set to any set $\mathcal{S}$ from Definition 2.8. An algorithm to compute an inner approximation of such a set is presented in [86, 32], which we call the approximate N-Step Robust Controllable Set.

Definition 3.1 (ROA of the Robust MPC) *The ROA for the proposed robust MPC, denoted by $\mathcal{R}$, is defined as the union of the N_t-Step Robust Controllable Sets to the terminal set $\mathcal{X}_N$ under the policy (3.10), for $N_t \in \{1, 2, \ldots, N\}$.*

An inner approximation to the ROA, which we call the approximate ROA, can be obtained using the approximate N_t-Step Robust Controllable Sets for $N_t \in \{1, 2, \dots, N\}$.

Assumption 3.3 *The matrices P and R in $\ell(x,u) = x^\top P x + u^\top R u$ are positive definite, i.e., $P \succ 0$ and $R \succ 0$.*

Assumption 3.4 *The matrix P_N which defines the terminal cost in (3.8) is chosen as a matrix $P_N \succ 0$ satisfying*

$$x^\top \Big(-P_N + (P + K^\top R K) + \bar{A}_{\mathrm{cl}}^\top P_N \bar{A}_{\mathrm{cl}} \Big) x \leq 0 \tag{3.15}$$

for all $x \in \mathcal{X}_N$, where $\bar{A}_{\mathrm{cl}} = \bar{A} + \bar{B}K$.

Definition 3.2 (ISS Lyapunov Function [87]) *Consider the closed-loop system given by*

$$x_{t+1} = Ax_t + Bu^\star_{t|t}(x_t) + w_t, \ \forall t \geq 0. \tag{3.16}$$

Then the origin is called Input to State Stable (ISS), with a ROA $\mathcal{R} \subset \mathbb{R}^n$, if there exists class-$\mathcal{K}_\infty$ functions $\alpha_1(\cdot)$, $\alpha_2(\cdot)$, $\alpha_3(\cdot)$, a class-$\mathcal{K}$ function $\sigma(\cdot)$ and a function $V(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}_{\geq 0}$ continuous at the origin, such that,

$$\begin{aligned} &\alpha_1(\|x\|) \leq V(x) \leq \alpha_2(\|x\|), \ \forall x \in \mathcal{R}, \\ &V(x_{t+1}) - V(x_t) \leq -\alpha_3(\|x_t\|) + \sigma(\|\tilde{w}_i\|_{\mathcal{L}_\infty}), \end{aligned}$$

where $\tilde{w}_i = \Delta_A^{\mathrm{tr}} x_i + \Delta_B^{\mathrm{tr}} u_i + w_i$ and $\|\tilde{w}_i\|_{\mathcal{L}_\infty} = \sup_{i \in \{0,\dots,t\}} \|\tilde{w}_i\|$. Function $V(\cdot)$ is called an ISS Lyapunov function for (3.16).

Theorem 3.2 *Let Assumptions 3.1-3.4 hold and let $x_0 \in \mathcal{R}$. Then, the optimal cost of (3.8) with (3.9), i.e., $J^\star(x_t)$ is an ISS Lyapunov function for the closed-loop system (3.16). This guarantees Input to State Stability of the origin of (3.16).*

Proof *From Assumption 3.3 we know that, $\alpha_1(\|x_t\|_2) \leq \ell(x,0) \leq J^\star(x_t)$ for some $\alpha_1(\cdot) \in \mathcal{K}_\infty$ and for all $x \in \mathcal{R}$. Moreover, since (3.8) can be reformulated into a parametric QP for each horizon length N_t, constraint set (3.1e) is compact, and $J^\star(0) = 0$, from [55, Theorem 23], we know $J^\star(x_t) \leq \alpha_2(\|x_t\|_2)$ for some $\alpha_2(\cdot) \in \mathcal{K}_\infty$ and for all $x_t \in \mathcal{R}$. We complete the proof by considering the same two cases :*

Case 1: *($N_t^\star = 1$) Consider the case of $N_t^\star = 1$. The optimal nominal cost at timestep t is written as*

$$\begin{aligned} J^\star(x_t) &= \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) + (\bar{x}^\star_{t+1|t})^\top P_N \bar{x}^\star_{t+1|t} \\ &\geq \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) + \ell(\bar{x}^\star_{t+1|t}, K\bar{x}^\star_{t+1|t}) + ((\bar{A} + \bar{B}K)\bar{x}^\star_{t+1|t})^\top P_N ((\bar{A} + \bar{B}K)\bar{x}^\star_{t+1|t}) \quad (3.17a) \\ &= \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) + q(\bar{x}^\star_{t+1|t}), \quad (3.17b) \end{aligned}$$

where in (3.17a) we have used Assumption 3.4, and at timestep $(t+1)$ the feasible input $\bar{u}_{t+1|t} = K\bar{x}^\star_{t+1|t}$ as discussed in (3.13). As (3.13) is a feasible policy at timestep $(t+1)$ with horizon length $N_{t+1} = 1$, the optimal cost of the MPC problem for any horizon length $N^\star_{t+1} = \{1, 2, \ldots, N\}$ can be bounded from above as:

$$J^\star(x_{t+1}) \leq \ell(\bar{x}_{t+1|t+1}, \bar{u}_{t+1|t}(\bar{x}_{t+1|t+1})) + Q(\bar{x}_{t+2|t+1}) = q(\bar{x}_{t+1|t+1}), \tag{3.18}$$

with $\bar{x}_{t+1|t+1} = \bar{x}^\star_{t+1|t} + \tilde{w}_t$, with $\tilde{w}_t = \Delta^{\text{tr}}_A x_t + \Delta^{\text{tr}}_B \bar{u}^\star_{t|t} + w_t$. Combining (3.17b)–(3.18) we obtain:

$$J^\star(x_{t+1}) - J^\star(x_t) \leq q(\bar{x}^\star_{t+1|t} + \tilde{w}_t) - \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) - q(\bar{x}^\star_{t+1|t}) \leq -\alpha_3(\|x_t\|_2) + L_q\|\tilde{w}_i\|_{\mathcal{L}_\infty}, \tag{3.19}$$

where $q(\cdot)$ is L_q-Lipschitz as it is a sum of quadratics in $\mathcal{X}$.

***Case 2:** ($N^\star_t \geq 2$) Now consider*

$$J^\star(x_t) = \sum_{k=t}^{t+N^\star_t-1} \ell(\bar{x}^\star_{k|t}, \bar{u}^\star_{k|t}) + Q(\bar{x}^\star_{t+N^\star_t|t}) = \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) + q(\bar{x}^\star_{t+1|t}), \tag{3.20}$$

where $\{\bar{x}^\star_{t|t}, \bar{x}^\star_{t+1|t}, \ldots, \bar{x}^\star_{t+N^\star_t|t}\}$ is the optimal predicted nominal trajectory under the optimal nominal input sequence $\{\bar{u}^\star_{t|t}, \bar{u}^\star_{t+1|t}, \ldots, \bar{u}^\star_{t+N^\star_t-1|t}\}$, where $\bar{u}^\star_{k|t} = u^\star_{k|t}(\bar{x}^\star_{k|t})$ for all $k \in \{t, t+1, \ldots, t+(N^\star_t-1)\}$. The quantity $q(\bar{x}^\star_{t+1|t})$ provides the total nominal cost from timestep $(t+1)$ to $(t+N^\star_t)$ under the following optimal control policy

$$\{u^\star_{t+1|t}(\cdot), \ldots, u^\star_{t+N^\star_t-1|t}(\cdot)\}. \tag{3.21}$$

We know that (3.14) is a feasible policy sequence for (3.8) at timestep $(t+1)$ with horizon length $N_{t+1} = (N^\star_t - 1)$. After $\bar{x}_{t+1} = x_{t+1}$ is obtained with closed-loop system evolution (3.16), with this feasible policy sequence (3.21), the optimal nominal cost of (3.8) at timestep $(t+1)$ for any $N^\star_{t+1} \in \{1, 2, \ldots, N\}$ can be bounded as:

$$J^\star(x_{t+1}) \leq \sum_{k=t+1}^{t+N^\star_t-1} \ell(\bar{x}_{k|t+1}, u^\star_{k|t}(\bar{x}_{k|t+1})) + Q(\bar{x}_{t+N^\star_t|t+1}) = q(\bar{x}_{t+1|t+1}), \tag{3.22}$$

where we have used the feasible nominal trajectory obtained with the policy (3.21), given as

$$\bar{x}_{k|t+1} = \bar{A}^{k-t-1}(\bar{A}x_t + \bar{B}u^\star_{t|t}(x_t) + \tilde{w}_t) + \sum_{i=t+1}^{k-1} \bar{A}^{k-1-i}\bar{B}u^\star_{i|t}(\bar{x}_{k|t+1}),$$

for $k = \{t+2, t+3, \ldots, t+N^\star_t\}$. Moreover, we know that

$$\bar{x}_{t+1|t+1} = \bar{x}^\star_{t+1|t} + \tilde{w}_t, \tag{3.23}$$

with $\tilde{w}_t = \Delta_A^{\text{tr}} x_t + \Delta_B^{\text{tr}} \bar{u}^\star_{t|t} + w_t$. *Combining* (3.20)–(3.23):

$$\begin{aligned} J^\star(x_{t+1}) - J^\star(x_t) &= q(\bar{x}^\star_{t+1|t} + \tilde{w}_t) - \ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) - q(\bar{x}^\star_{t+1|t}) \\ &\leq -\ell(\bar{x}^\star_{t|t}, \bar{u}^\star_{t|t}) + L_q \|\tilde{w}_t\| \leq -\ell(\bar{x}^\star_{t|t}, 0) + L_q \|\tilde{w}_t\| \\ &\leq -\alpha_3(\|x_t\|_2) + L_q \|\tilde{w}_i\|_{\mathcal{L}_\infty}. \end{aligned} \tag{3.24}$$

Combining (3.19) *and* (3.24)*, the origin of* (3.16) *is ISS according to Definition 3.2.*

3.4 Numerical Simulations

We choose $N = 5$ and compute approximate solutions to the problem (2.54). The feedback gain K satisfying Assumption 3.1 is chosen as $K = -[0.4866, 0.4374]$. The source code is available at https://github.com /monimoyb/RMPC_SimpleTube.

3.4.1 Comparison with [15]

The tube cross section (Z) is chosen as the minimal robust positive invariant set for system (2.10) under a feedback $u = -[0.7701, 0.7936]x$, and the terminal set ($\mathcal{X}_f$) is chosen as $\mathcal{X}_N$ in (3.5). See [15] for details on these quantities. We then choose a set of $N_{\text{init}} = 100$ initial states x_S, created by a 10×10 uniformly spaced grid of the set of state constraints. From each of these initial state samples we check the feasibility of the tube MPC problem in [15, Section 5]. The code to solve the tube MPC is used from [88]. The convex hull of the feasible initial states (largest out of horizons $N \leq 5$) inner approximates the ROA of the tube MPC. This is compared to the approximate ROA of our proposed robust MPC. The comparison is shown in Fig. 3.1. The approximate ROA from our approach is about 1.05x *larger* in volume, but containing 98% of that of the tube MPC[2]. However, for any $N \leq 5$, the tube MPC needs higher computation times than for all $N_t \in \{1, 2, \ldots, N\}$ combined in our approach. This is shown in Table 3.1.

Table 3.1: Comparison of avg. online computation times [sec].

Horizon	**Proposed Robust MPC**	**Tube MPC in [15]**
$N_t = 1$	0.0026	0.0062
$N_t = 2$	0.0023	0.0753
$N_t = 3$	0.0038	0.1612
$N_t = 4$	0.0056	0.2556
$N_t = 5$	0.0078	0.3384

[2]Note, the tube MPC in [15, Section 5] has no recursive feasibility guarantees.

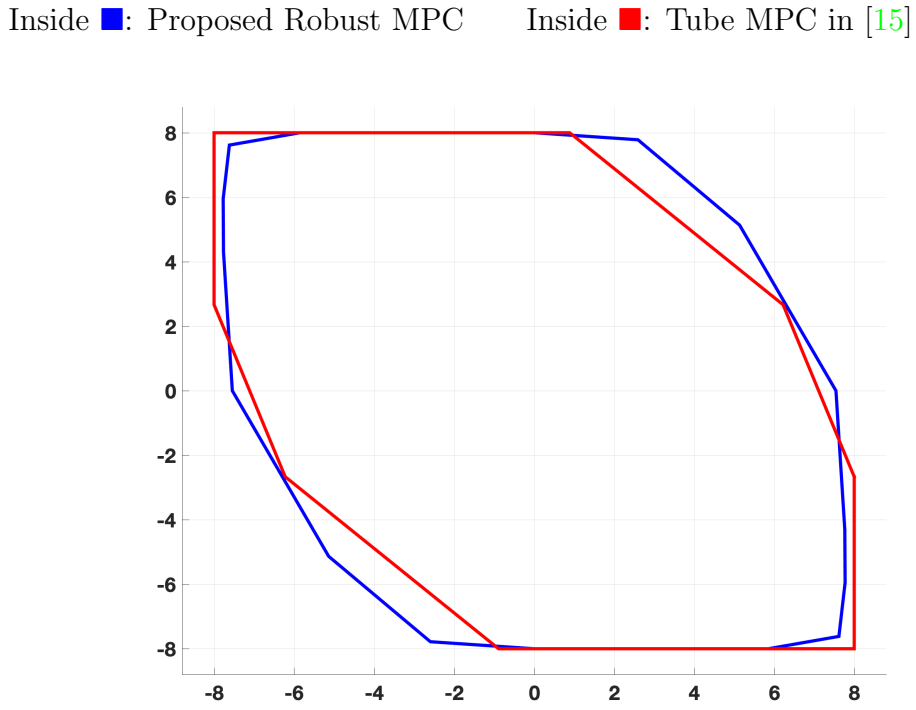


Figure 3.1: Comparison of the approximate ROA of the proposed robust MPC with $N = 5$ and the approximate ROA of the tube MPC in [15, Section 5].

3.4.2 Comparison with [19]

Recall (2.54) and Remark 2.7. We show the comparison of the approximate ROA of our proposed robust MPC and the ellipsoidal ROA of [19] in Fig. 3.2. The ROA of our proposed approach is is about 28% larger than the corresponding ROA of [19]. The computation times are compared in Table 3.2. Due to Remark 2.7, the associated online times of [19] are independent of the horizon, as it only solves an LMI for obtaining the ellipsoidal ROA. The offline time of our proposed algorithm is the time taken to compute the terminal set. On the other hand, the large offline computation time in [19] is due to a logarithmic search required for optimizing a parameter (denoted by α in [19]). We can conclude from Fig. 3.2 and Table 3.2 that our proposed algorithm in this chapter obtains a better computation complexity vs conservatism trade-off over the tube MPC in [19], considering both online and offline computation times.

3.4.3 Roll-Out Alternative and Comparison with [84]

A computationally cheaper alternative can be obtained as follows: Once an optimization problem in (3.8) at timestep $t = 0$ is feasible for some horizon length $N_0 = \bar{N}_0 \in \{1, 2, \ldots, N\}$, the corresponding optimal policy sequence: $\{u^\star_{0|0}, u^\star_{1|0}(\cdot), \ldots, u^\star_{\bar{N}_0-1|0}(\cdot)\}$ can be used to ob-

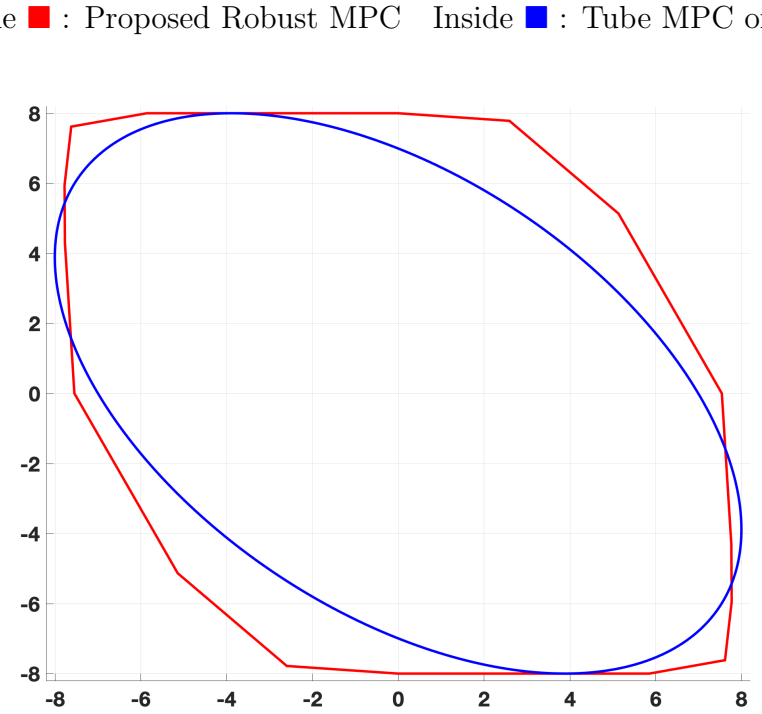


Figure 3.2: Comparison of the approx. ROA of the proposed MPC with $N = 5$ and the ROA of [19].

Table 3.2: Comparison of average computation times [sec].

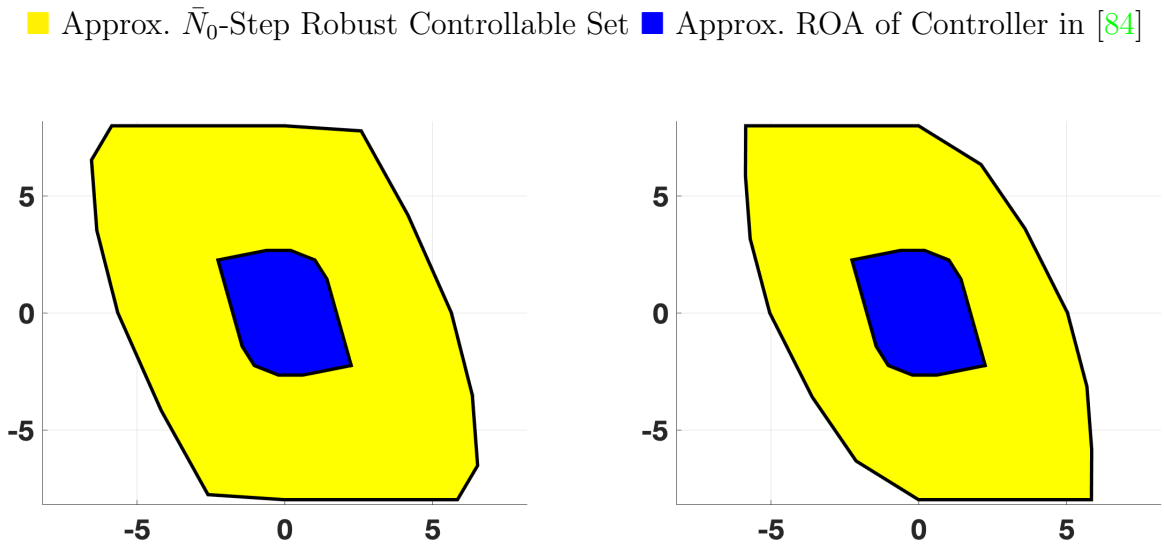
Horizon	**Proposed Robust MPC**		**Tube MPC in [19]**	
	online	**offline**	**online**	**offline**
$N_t = 1$	0.0026	7.12	0.03	53.08
$N_t = 2$	0.0023	7.12	0.03	53.08
$N_t = 3$	0.0038	7.12	0.03	53.08
$N_t = 4$	0.0056	7.12	0.03	53.08
$N_t = 5$	0.0078	7.12	0.03	53.08

tain a safe open-loop policy for all timesteps as:

$$\Pi_{\text{ol}}^{\text{safe}}(x_t) = \begin{cases} u^\star_{t|0}(x_t), & \text{if } t \leq (\bar{N}_0 - 1), \\ Kx_t, & \text{otherwise.} \end{cases} \tag{3.25}$$

Policy (3.25) maintains the robust satisfaction of (3.1e) for all timesteps, without re-solving (3.8). From each of the previous 100 initial state samples, we now check the feasibility of the

constrained LQR synthesis problem in [84, Section 2.3]. We pick the FIR length (same as control horizon length) as $L = 15$, with $\tau = 0.99$ and $\tau_\infty = 0.2$. See [84, Problem 2.8] for details on these parameters. The comparison of the approximate $\bar{N}_0$-Step Robust Controllable Sets and the approximate region of attraction of the algorithm of [84, Section 2.3] is shown in Fig. 3.3. The volumes of the approximate $\bar{N}_0$-Step Robust Controllable Sets are bigger than the approximate ROA of the controller in [84, Section 2.3] for all $\bar{N}_0 \leq 5$, showing that the roll-out policy (3.25) yields up to approximately 12x lower conservatism.



(a) $\bar{N}_0 = 2, 3, 4$ (comparable sets) (b) $\bar{N}_0 = 5$

Figure 3.3: A safe open-loop policy (3.25) is guaranteed to exist at all times with initial states in the yellow regions.

3.5 Chapter Summary

We proposed a computationally efficient approach to design a robust MPC for constrained uncertain linear parameter varying systems. The uncertainty considered included both mismatch in the system dynamics matrices, and an additive disturbance. The designed MPC is recursively feasible and the origin of the closed-loop system is Input to State stable. With detailed numerical simulations and comparisons with [15, 19, 27], we demonstrated that the proposed approach can be a simple and viable alternative to balance the trade-off between computational complexity and conservatism in robust MPC design.

Appendix

As shown in the appendix of [55], matrices $\bar{\mathbf{A}}$, $\mathbf{C}$ and $\mathbf{G}$ for a horizon $\bar{N}$ are:

$$\begin{aligned}
\mathbf{G} &= I_{n\bar{N}} + \sum_{k=1}^{\bar{N}-1} \mathcal{L}_{\bar{N}}^{k} \otimes \bar{A}^{k}, \\
\bar{\mathbf{A}} &= \mathrm{diag}(\bar{A}, \bar{A}^{2}, \ldots, \bar{A}^{\bar{N}-1}), \\
\mathbf{C} &= \mathbf{G} \cdot (I_{\bar{N}} \otimes \bar{B}),
\end{aligned}$$

with $\mathcal{L}$ being the lower shift matrix, where the operation $A \otimes B$ denotes the Kronecker product of two matrices A and B.

Chapter 4

Robust MPC with Optimization-Based Constraint Tightening

This chapter is based on the published work [89]. Instead of using the worst-case constraint tightening tubes around any predicted nominal trajectory as done in Chapter 3, in this chapter we propose an optimization-based constraint tightening strategy which is a function of decision variables in the control synthesis problem. This lowers the conservatism in the proposed control design approach, while keeping it computationally viable for online control synthesis.

4.1 Summary of Contributions

The key contributions of this chapter are summarized as:

- We propose a novel constraint tightening strategy which is decoupled into two phases. In the first phase, we bound the effect of model uncertainty on any predicted *nominal* (i.e., uncertainty free) trajectory. These bounds are computed offline. This phase is motivated by [27, 26]. In the second phase, the MPC is designed utilizing the above bounds, so that the constraint tightenings are functions of decision variables in the control synthesis problem. This second phase is motivated by tube MPC works such as [15, 58, 76, 18, 19].
- We solve a tractable convex optimization problem online using a shrinking horizon approach for the MPC. With an appropriately constructed terminal set and a terminal cost, we prove robust satisfaction of the imposed constraints by the closed-loop system, and Input to State stability of the origin.

4.2 Robust MPC Design

We consider the linear systems of the form (2.10). The major drawback of the robust MPC approach presented in the previous chapter is the bound (2.57). As pointed out in [80], this bound can be rather conservative. To alleviate this issue, in this chapter we present the design of an alternative robust MPC where the constraint tightenings are chosen as functions of decision variables, instead of using the worst case bounds in (2.57).

4.2.1 Predicted State Evolution

We first denote the sequences of vectors:

$$\begin{aligned} \mathbf{u}_t &= [u_{t|t}^\top, u_{t+1|t}^\top(\cdot), \dots, u_{t+N-1|t}^\top(\cdot)]^\top, \\ \bar{\mathbf{x}}_t &= [\bar{x}_{t|t}^\top, \bar{x}_{t+1|t}^\top, \dots, \bar{x}_{t+N-1|t}^\top]^\top. \end{aligned} \tag{4.1}$$

In this section, we use the following two observations: First, keeping the nominal state trajectory $\bar{\mathbf{x}}_t$ as a decision variable in the MPC problem (3.1) maintains certain structure that can be exploited to bound the effect of model uncertainty on a predicted nominal trajectory, similar to [26, 27]. And second, the predicted nominal trajectory and its associated inputs along the horizon are computed by reformulating (3.1) and solving a robust optimization problem, similar to tube MPC approaches such as [15, 58, 76, 18, 19]. We thus attempt to merge the benefits of both these ideas in this work.

Recall the nominal system dynamics from (3.1b) given as $\bar{x}_{t+1} = \bar{A}\bar{x}_t + \bar{B}\bar{u}_t$, with $\bar{u}_t = u_t(\bar{x}_t)$. Denote the vectors $\mathbf{x}_t, \mathbf{w}_t \in \mathbb{R}^{nN}$ and $\Delta\mathbf{u}_t \in \mathbb{R}^{mN}$ as:

$$\begin{aligned} \mathbf{x}_t &= \begin{bmatrix} x_{t+1|t}^\top & x_{t+2|t}^\top & \dots & x_{t+N|t}^\top \end{bmatrix}^\top, \\ \mathbf{w}_t &= \begin{bmatrix} w_{t|t}^\top & w_{t+1|t}^\top & \dots w_{t+N-1|t}^\top \end{bmatrix}^\top, \\ \Delta\mathbf{u}_t &= \begin{bmatrix} \Delta u_{t|t}^\top & \Delta u_{t+1|t}^\top(\cdot) & \dots & \Delta u_{t+N-1|t}^\top(\cdot) \end{bmatrix}^\top, \end{aligned} \tag{4.2}$$

where $\Delta u_{k|t}(\cdot) = u_{k|t}(\cdot) - \bar{u}_{k|t}$ for $k \in \{t, t+1, \dots, t+N-1\}$. Using (4.1) and (4.2), we can write the state evolution along the prediction horizon as:

$$\mathbf{x}_t = \mathbf{A}^x\bar{\mathbf{x}}_t + \mathbf{A}^u\mathbf{u}_t + \mathbf{A}^{\Delta u}\Delta\mathbf{u}_t + \mathbf{A}^w\mathbf{w}_t, \tag{4.3}$$

where $\mathbf{x}_t$ denotes the prediction of possible evolutions of the realized states[1], and in (4.3) the predicted nominal states along the horizon, i.e., $\bar{\mathbf{x}}_t$ from (4.1) appears directly and *not* expressed in terms of $\{x_t, u_{t|t}, u_{t+1|t}(\cdot), \dots, u_{t+N-1|t}(\cdot)\}$, as in [55]. The prediction dynamics matrices $\mathbf{A}^x, \mathbf{A}^u, \mathbf{A}^{\Delta u}$ and $\mathbf{A}^w$ in (4.3) depend on $\bar{B}, \Delta_A, \Delta_B$ and $(\bar{A}+\Delta_A), (\bar{A}+\Delta_A)^2, \dots, (\bar{A}+\Delta_A)^{N-1}$. We define $A_\Delta = \bar{A} + \Delta_A$ for some possible $\Delta_A \in \mathcal{P}_A$. Then $A_\Delta \in \mathcal{P}_{A_\Delta}$, with the set $\mathcal{P}_{A_\Delta}$ defined as:

$$\mathcal{P}_{A_\Delta} = \{A_m : A_m = \bar{A} + \Delta_A,\ \Delta_A \in \mathcal{P}_A\}. \tag{4.4}$$

[1]Note, (4.2) implies (4.3) is a compact state update equation.

Using (4.4) we rewrite the matrices in (4.3) as follows:

$$\begin{aligned}\mathbf{A}^x &= \bar{\mathbf{A}} + \Big(\bar{\mathbf{A}}_1 + \mathbf{A}_\delta\Big)\boldsymbol{\Delta}_A,\\ \mathbf{A}^u &= \bar{\mathbf{B}} + \Big(\bar{\mathbf{A}}_1 + \mathbf{A}_\delta\Big)\boldsymbol{\Delta}_B,\\ \mathbf{A}^{\Delta u} &= \Big(\bar{\mathbf{A}}_1 - \mathbf{I}_n + \mathbf{A}_\delta\Big)\bar{\mathbf{B}}, \text{ and}\\ \mathbf{A}^w &= \mathbf{I}_n + \bar{\mathbf{A}}_v\mathbf{A}_\Delta,\end{aligned} \tag{4.5}$$

where $\mathbf{I}_n = (I_N \otimes I_n) \in \mathbb{R}^{nN\times nN}$, $\bar{\mathbf{A}} = (I_N \otimes \bar{A}) \in \mathbb{R}^{nN\times nN}$, $\bar{\mathbf{B}} = (I_N \otimes \bar{B}) \in \mathbb{R}^{nN\times mN}$, $\boldsymbol{\Delta}_A = (I_N \otimes \Delta_A) \in \mathbb{R}^{nN\times nN}$, and $\boldsymbol{\Delta}_B = (I_N \otimes \Delta_B) \in \mathbb{R}^{nN\times mN}$. The matrices $\bar{\mathbf{A}}_1$, $\mathbf{A}_\delta$, $\bar{\mathbf{A}}_v$ and $\mathbf{A}_\Delta$ are defined in 4.7.1 in the Appendix. Matrices $\mathbf{A}_\delta$ and $\mathbf{A}_\Delta$ depend on parametric uncertainty matrices Δ_A and Δ_B. In the next sections, we substitute the matrices from (4.5) in (4.3) in order to design a control policy that robustly satisfies (3.1e)-(3.1f) along the horizon.

4.2.2 The Novel Optimization-Based Constraint Tightening

The terminal set $\mathcal{X}_N$ in (3.1f) is defined by $\mathcal{X}_N = \{x : H_N^x x \le h_N^x\}$, with $H_N^x \in \mathbb{R}^{r_N\times n}, h_N^x \in \mathbb{R}^{r_N}$. We denote the matrices $\mathbf{F}^x = \text{diag}(I_{N-1} \otimes H^x, H_N^x) \in \mathbb{R}^{(r(N-1)+r_N)\times nN}$, and $\mathbf{f}^x = (h^x, h^x, \dots, h_N^x) \in \mathbb{R}^{r(N-1)+r_N}$ for any given N. Using (4.3), the robust state constraints in (3.1) for predicted states along the prediction horizon and at the end of the horizon can then be written as:

$$\mathbf{F}^x\mathbf{x}_t \le \mathbf{f}^x,\ \forall\Delta_A \in \mathcal{P}_A,\ \forall\Delta_B \in \mathcal{P}_B,\ \forall w_t \in \mathbb{W}. \tag{4.6}$$

We guarantee satisfaction of (4.6) using the following: Suppose for any a, b, we need to guarantee $a \le b$. We first obtain an upper bound c, such that $a \le c$, and then we impose $c \le b$. This is a sufficient condition for $a \le b$. Accordingly, using (4.3) and (4.5) constraint (4.6) for all timesteps $t \ge 0$ can be replaced row-wise as:

$$\begin{aligned}&\mathbf{F}_i^x((\bar{\mathbf{A}} + \bar{\mathbf{A}}_1\boldsymbol{\Delta}_A)\bar{\mathbf{x}}_t + (\bar{\mathbf{B}} + \bar{\mathbf{A}}_1\boldsymbol{\Delta}_B)\mathbf{u}_t + \cdots\\ &\quad + (\bar{\mathbf{A}}_1 - \mathbf{I}_n)\bar{\mathbf{B}}\Delta\mathbf{u}_t + \mathbf{w}_t) + \mathbf{t}_1^i\|\bar{\mathbf{x}}_t\| + \mathbf{t}_2^i\|\mathbf{u}_t\| + \mathbf{t}_3^i\|\Delta\mathbf{u}_t\| + \mathbf{t}_w^i\|\mathbf{w}_t\| \le \mathbf{f}_i^x,\\ &\forall\Delta_A \in \mathcal{P}_A,\ \forall\Delta_B \in \mathcal{P}_B,\ \forall w_t \in \mathbb{W},\end{aligned} \tag{4.7}$$

for $i \in \{1, 2, \dots, r(N-1) + r_N\}$, where recall that r and r_N are the number of rows of H^x and H_N^x, respectively. In Appendix 4.7.2 we detail the derivation of (4.7) from (4.6) and the computation of the bounds $\{\mathbf{t}_w^i, \mathbf{t}_1^i, \mathbf{t}_2^i, \mathbf{t}_3^i\}$ for rows $i \in \{1, 2, \dots, r(N-1)+r_N\}$. In (4.7) we have bounded the effect of model mismatch, i.e., the matrices $\mathbf{A}_\delta, \mathbf{A}_\Delta, \boldsymbol{\Delta}_A, \boldsymbol{\Delta}_B$ on predicted nominal states. These bounds, denoted as $\{\mathbf{t}_w^i, \mathbf{t}_1^i, \mathbf{t}_2^i, \mathbf{t}_3^i\}$ for rows $i \in \{1, 2, \dots, r(N-1)+r_N\}$, are computed *offline*, and are derived in detail in (4.27)-(4.31) in the Appendix, where we also show that (4.7) is sufficient for (4.6).

In constraint (4.7), note that the decision variables are the predicted nominal trajectory $\bar{\mathbf{x}}_t$, and the sequence of input policies $\mathbf{u}_t$. These decision variables multiply effects of the

bounds $\mathbf{t}_1^i, \mathbf{t}_2^i$ and $\mathbf{t}_3^i$. In conclusion, the tightening of the original constraint (3.1e) proposed in (4.7) depends on the optimization variables, $\bar{\mathbf{x}}_t$, $\mathbf{u}_t$, and $\Delta\mathbf{u}_t$. This is a key contribution of our work. Alternatively in [27, 26], the constraint tightening is obtained bounding the closed-loop system response, which involves the norm of the product between the decision variables and the uncertainty. Therefore the method in [27, 26] needs to resort to a grid search over parameters to obtain sufficient conditions for satisfying (3.1e) robustly. Tube MPC methods such as [15, 76, 18, 19] could lead to tightenings equivalent to (4.7) under appropriately chosen parametrization of tube cross sections. However, the computation times of these approaches can increase noticeably with horizon length, primarily due to an increase in the number of constraints in the MPC problem. See Section 2.9.2, Section 2.10.1 and [75, Chapter 5] for additional details.

4.2.3 Tractable MPC Problem with Safe Backup

In this section we present the MPC reformulation of (3.1) which guarantees robust constraint satisfaction at all timesteps $t \geq 0$, and Input to State Stability of the origin.

Uncertainty Representation Mismatch

We start with the following observation: The terminal set $\mathcal{X}_N$ from (3.5) is robustly invariant to all uncertainty of the form: $\forall \Delta_A \in \mathcal{P}_A,\ \forall \Delta_B \in \mathcal{P}_B,\ \forall w \in \mathbb{W},\ \forall t \geq 0$, when the state feedback policy $\kappa_N(x) = Kx$ is used in (2.10). However, along the prediction horizon we use bounds $\{\mathbf{t}_w^i, \mathbf{t}_1^i, \mathbf{t}_2^i, \mathbf{t}_3^i\}$, which are obtained by more conservative tightenings from Hölder's and triangle inequalities, and induced norm consistency and submultiplicativity properties (see (4.27)-(4.31) in the Appendix). Thus the uncertainty bounds along the horizon over-approximate the effect of the true uncertainty used to compute the terminal set. This implies that the classical shifting argument [6, Chapter 12] for recursive MPC feasibility cannot be used in this case.

The Use of Shrinking Horizons

As a consequence, to ensure robust satisfaction of constraints (3.1e) by system (2.10) at all timesteps and Input to State Stability of the origin, we will use the following strategy: (i) at any given timestep, we solve the MPC reformulation of problem (3.1) in a shrinking horizon fashion, i.e., we choose the MPC horizon length at timestep t, denoted by N_t, as:

$$N_t = \begin{cases} N - t, \text{ if } t \in \{0, 1, \dots N-2\}, \\ 1, \text{ otherwise.} \end{cases} \tag{4.8}$$

If the shrinking horizon MPC problem is infeasible, we use the time-shifted optimal policy from a previous timestep as a safe backup policy to guarantee robust satisfaction of (3.1e), and (ii) we design the terminal cost matrix P_N so that the MPC open-loop cost is a Lyapunov function inside $\mathcal{X}_N$. This design choice, together with the shrinking horizon strategy, which

guarantees finite time convergence to $\mathcal{X}_N$, allows us to show Input to State Stability of the origin.

Input Policy, Terminal Set, MPC Problem

We use the input policy parametrization from (2.27). Then the sequence of predicted inputs can be written as $\mathbf{u}_t = \mathbf{M}_t^{(N)}\mathbf{w}_t + \bar{\mathbf{u}}_t^{(N)}$ at timestep t, where $\mathbf{M}_t^{(N)} \in \mathbb{R}^{mN\times nN}$ and $\bar{\mathbf{u}}_t^{(N)} \in \mathbb{R}^{mN}$ are shown in (2.28). The terminal set $\mathcal{X}_N$ is constructed following Section 3.2.2. We introduce the following set of required notations. Denote the set $\mathbb{W} = \{w \in \mathbb{R}^n : H^w w \leq h^w\}$ with $H^w \in \mathbb{R}^{a\times n}$ and $h^w \in \mathbb{R}^a$. For a horizon length of N_t from (4.8), this gives $\mathbf{W} = \{\mathbf{w} \in \mathbb{R}^{nN_t} : \mathbf{H}^w\mathbf{w} \leq \mathbf{h}^w\}$, with $\mathbf{H}^w = I_{N_t} \otimes H^w \in \mathbb{R}^{aN_t\times nN_t}$ and $\mathbf{h}^w = (h^w, h^w, \dots, h^w) \in \mathbb{R}^{aN_t}$. Also denote the matrices $\mathbf{H}^u = I_{N_t} \otimes H^u \in \mathbb{R}^{oN_t\times mN_t}$, and $\mathbf{h}^u = (h^u, h^u, \dots, h^u) \in \mathbb{R}^{oN_t}$. Moreover, we denote vectors $\mathbf{t}_j^{(N_t)} = [\mathbf{t}_j^1, \mathbf{t}_j^2, \dots, \mathbf{t}_j^{r(N_t-1)+r_N}]^\top$ for the indices $j \in \{w, 1, 2, 3\}$. We use the notation $\bar{\mathbf{x}}_t^{(N_t)}$ for each horizon length N_t, to explicitly indicate the varying dimension of the vector $\bar{\mathbf{x}}_t$ previously introduced in (4.2).

In (4.7) the input policy was not specified. We now use policy parametrization (3.3) in (4.7) and consider the following two cases[2]:

Case 1: ($N_t \geq 2$, i.e., $t \leq N-2$)

$$\max_{\mathbf{w}_t\in\mathbf{W}} \mathbf{F}^x\Big(\bar{\mathbf{A}}\bar{\mathbf{x}}_t^{(N_t)} + \bar{\mathbf{B}}(\mathbf{M}_t^{(N_t)}\mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)}) + (\bar{\mathbf{A}}_1 - \mathbf{I}_n)\bar{\mathbf{B}}\mathbf{M}_t^{(N_t)}\mathbf{w}_t + \mathbf{w}_t\Big) \leq \mathbf{f}_{\text{tight}}^x, \tag{4.9a}$$

Case 2: ($N_t = 1$, i.e., $t \geq N-1$)

$$\max_{\substack{w_t\in\mathbb{W}\\ \Delta_A\in\mathcal{P}_A\\ \Delta_B\in\mathcal{P}_B}} H_N^x((\bar{A}+\Delta_A)\bar{\mathbf{x}}_t^{(1)} + (\bar{B}+\Delta_B)\bar{\mathbf{u}}_t^{(1)} + w_t) \leq h_N^x, \tag{4.9b}$$

for $N_t \in \{1, 2, \dots, N\}$. The tightened set of constraints $\mathbf{f}_{\text{tight}}^x$ are given by

$$\mathbf{f}_{\text{tight}}^x = \mathbf{f}^x - \mathbf{t}_{\delta 1}^{(N_t)}\|\bar{\mathbf{x}}_t^{(N_t)}\| - \mathbf{t}_{\delta 3}^{(N_t)}\|\mathbf{M}_t^{(N_t)}\|_p\mathbf{w}_{\max} - \mathbf{t}_{\delta 2}^{(N_t)}\|\bar{\mathbf{u}}_t^{(N_t)}\| - \mathbf{t}_w^{(N_t)}\mathbf{w}_{\max}, \tag{4.10}$$

with $\|\mathbf{w}_t\| \leq \mathbf{w}_{\max}$ for all $t \geq 0$, where

$$\begin{aligned} \mathbf{t}_{\delta 1}^{(N_t)} &= \mathbf{t}_{\delta A}^{(N_t)} + \mathbf{t}_1^{(N_t)},\ \mathbf{t}_{\delta 2}^{(N_t)} = \mathbf{t}_{\delta B}^{(N_t)} + \mathbf{t}_2^{(N_t)}, \\ \mathbf{t}_{\delta 3}^{(N_t)} &= \mathbf{t}_{\delta B}^{(N_t)} + \mathbf{t}_2^{(N_t)} + \mathbf{t}_3^{(N_t)}, \end{aligned} \tag{4.11}$$

using the bounds

$$\max_{\Delta_A\in\mathcal{P}_A} \|\mathbf{F}_i^x\bar{\mathbf{A}}_1\boldsymbol{\Delta}_A\|_* = \mathbf{t}_{\delta A}^{(N_t),i}, \tag{4.12a}$$

$$\max_{\Delta_B\in\mathcal{P}_B} \|\mathbf{F}_i^x\bar{\mathbf{A}}_1\boldsymbol{\Delta}_B\|_* = \mathbf{t}_{\delta B}^{(N_t),i}, \tag{4.12b}$$

[2]The dimensions of $\mathbf{F}^x, \mathbf{f}^x, \bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \mathbf{I}_n$ and $\mathbf{w}_t$ vary depending on N_t. We omit showing this explicitly for brevity.

for $i \in \{1, 2, \ldots, r(N_t - 1) + r_N\}$. See 4.7.5 in the Appendix for a derivation of (4.9)-(4.10) from (4.7) using the bounds (4.11). Having formulated the state constraints, the input constraints in (3.1e) along the horizon can be written as:

$$\max_{\mathbf{w}_t \in \mathbf{W}} \mathbf{H}^u \left(\mathbf{M}_t^{(N_t)} \mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)} \right) \leq \mathbf{h}^u, \tag{4.13}$$

for $N_t \in \{1, 2, \ldots, N\}$. Using (4.9)-(4.13), at any timestep t we then solve

$$\begin{aligned} V_{t \to t+N_t}^{\mathrm{MPC}}(x_t, \mathbf{t}^{(N_t)}, N_t) := \\ \min_{\mathbf{M}_t^{(N_t)}, \mathbf{z}_t^{(N_t)}} \quad & (\mathbf{z}_t^{(N_t)})^\top \bar{Q}^{(N_t)} \mathbf{z}_t^{(N_t)} \\ \text{s.t.,} \quad & G_{\mathrm{eq}}^{(N_t)} \mathbf{z}_t^{(N_t)} = b_{\mathrm{eq}}^{(N_t)} x_t, \\ & (4.9\text{b}), (4.13) \text{ if } N_t = 1, \\ & (4.9\text{a}), (4.13) \text{ if } N_t > 1, \\ & \bar{x}_{t|t} = x_t, \end{aligned} \tag{4.14}$$

where we have denoted

$$\begin{aligned} \mathbf{z}_t^{(N_t)} &= \begin{bmatrix} (\bar{\mathbf{x}}_t^{(N_t)})^\top & \bar{x}_{t+N_t|t}^\top & (\bar{\mathbf{u}}_t^{(N_t)})^\top \end{bmatrix}^\top, \\ \mathbf{t}^{(N_t)} &= \{\mathbf{t}_w^{(N_t)}, \mathbf{t}_1^{(N_t)}, \mathbf{t}_2^{(N_t)}, \mathbf{t}_3^{(N_t)}\}, \\ \bar{Q}^{(N_t)} &= \mathrm{diag}(I_{N_t} \otimes P, P_N, I_{N_t} \otimes R), \\ G_{\mathrm{eq}}^{(N_t)} &= \begin{bmatrix} I_n & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\bar{A} & I_n & 0 & \cdots & 0 & 0 & -\bar{B} & 0 & \cdots & 0 \\ 0 & -\bar{A} & I_n & \cdots & 0 & 0 & 0 & -\bar{B} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\bar{A} & I_n & 0 & 0 & \cdots & -\bar{B} \end{bmatrix}, \\ b_{\mathrm{eq}}^{(N_t)} &= \begin{bmatrix} I_n \\ \mathbf{0}_{nN_t \times n} \end{bmatrix}. \end{aligned}$$

Note, we consider $\mathbf{t}^{(1)} = 0$. We solve problem (4.14) utilizing duality of convex programs. This is detailed in 4.7.6 in the Appendix. The constraint tightenings in (4.14) used in the robust state constraints are functions of the decision variables. This is our key contribution.

We assume that (4.14) is feasible at timestep $t = 0$ with $N_0 = N$. For $t \geq 1$, we apply the following policy

$$u_t^{\mathrm{MPC}}(x_t) = \begin{cases} \bar{u}_{t|t}^\star, \text{ if (4.14) is feasible,} \\ u_{t|t_{\mathrm{f}}}^\star(x_t), \text{ otherwise} \end{cases} \tag{4.15}$$

to system (2.10), where $t_{\mathrm{f}} \in \{0, 1, \ldots, N-1\}$ is the latest timestep where (4.14) was feasible previously. Thus, the time-shifted optimal policy from a previous timestep is utilized as a

safe backup, in case (4.14) loses feasibility. As we cannot measure w_t due to the presence of matrix uncertainties in (2.10), see [55, Section 5] for how to obtain the backup policy in state feedback form required for implementation. We then resolve (4.14) at the next timestep $(t+1)$ for horizon lengths N_{t+1} obtained from (4.8). The control algorithm is summarized in Algorithm 1.

Algorithm 1 Robust MPC with Optimization-Based Constraint Tightening

Inputs: $x_t, N, \mathbb{W}, \mathcal{X}_N, \mathbf{t}^{(N_t)}, \forall N_t \in \{2,3,\dots,N\}$
Initialize: $t=0$
while $t \geq 0$ **do**
  Set horizon length N_t from (4.8);
  Solve MPC problem (4.14);
  Apply closed-loop input (4.15) to (2.10);
  Set $t = t+1$;
end while

Remark 4.1 *Recall (2.12)–(2.13). One may also efficiently enumerate all possible vertex sequences of $\mathbf{\Delta}_A$ and $\mathbf{\Delta}_B$ for robustifying the term $\mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_A \bar{\mathbf{x}}_t^{(N_t)} + \mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_B \bar{\mathbf{u}}_t^{(N_t)}$ in (4.7) (with policy (3.3)). This partially replaces the bounds (4.12) to lower conservatism. As we use the backup policy in (4.15) without requiring recursive feasibility of (4.14), the number of such sequences is limited to the number of vertices characterizing the uncertain matrices (i.e., each vertex repeated N_t times along the horizon), and is not combinatorial. See [50, Figure 3] for further insights into why combinatorial enumerations are required otherwise.*

4.3 Robust Constraint Satisfaction and Stability

We first prove the robust satisfaction of constraints (3.1e) for the closed-loop system (2.10) and (4.15). Afterwards, we show the stability properties of the proposed robust MPC in Algorithm 1.

4.3.1 Feasibility of Robust Constraints

Theorem 4.1 *Let optimization problem (4.14) with tightened constraints (4.10) be feasible at timestep $t=0$ for $N_t = N$, where the bounds $\{\mathbf{t}_w^{(N_t)}, \mathbf{t}_1^{(N_t)}, \mathbf{t}_2^{(N_t)}, \mathbf{t}_3^{(N_t)}\}$ are obtained by solving (4.27)-(4.31). Then, the closed-loop system (2.10) and (4.15) robustly satisfies state and input constraints (3.1e), for all $t \geq 0$.*

Proof *By assumption, at timestep $t=0$ problem (4.14) with tightened constraints (4.10) is feasible, with a horizon length $N_t = N$. We then prove robust satisfaction of (3.1e) at all timesteps $t \geq 0$ with controller (4.15) in closed-loop, by considering the following two cases:*

***Case 1**: (2 ≤ N_t < N, i.e., 0 < t ≤ $N-2$) As (4.14) is feasible at timestep $t=0$ for $N_t = N$ chosen as per (4.8), let the corresponding optimal policy sequence be*

$$\{u^\star_{0|0}, u^\star_{1|0}(\cdot), \ldots, u^\star_{N-1|0}(\cdot)\}. \tag{4.16}$$

For timesteps $t \in \{1, 2, \ldots, N-2\}$ recall that we define the MPC policy in (4.15) as:

$$u_t^{\mathrm{MPC}}(x_t) = \begin{cases} \bar{u}^\star_{t|t}, \text{ if (4.14) is feasible,} \\ u^\star_{t|t_\mathrm{f}}(x_t), \text{ otherwise,} \end{cases} \tag{4.17}$$

where $t_\mathrm{f} \in \{0, 1, \ldots, N-1\}$ is the latest timestep when (4.14) was feasible. Policy (4.17) satisfies (3.1e) robustly for all $t \in \{1, 2, \ldots, N-2\}$, as it is a solution to the constrained robust optimal control problem (4.14). Moreover, from (4.16), we have that $t_\mathrm{f} = 0$ is a guaranteed certificate, in case (4.14) continues to be infeasible for all $t \in \{1, 2, \ldots, N-2\}$.

***Case 2**: ($N_t = 1$, i.e., $t \geq N-1$) Consider the timestep $t = N-1$, where from (4.8) the MPC horizon length $N_t = 1$. In this case we consider the constraints (4.9b):*

$$\max_{\substack{w_t \in \mathbb{W} \\ \Delta_A \in \mathcal{P}_A \\ \Delta_B \in \mathcal{P}_B}} H_N^x((\bar{A}+\Delta_A)\bar{\mathbf{x}}_t^{(1)} + (\bar{B}+\Delta_B)\bar{\mathbf{u}}_t^{(1)} + w_t) \leq h_N^x. \tag{4.18}$$

From (4.17) we know that at timestep $t = N-1$, there exists a t_f such that control action $u^\star_{t|t_\mathrm{f}}(x_t)$ robustly steers the state x_t to $\mathcal{X}_N$ in one timestep. Now, at $t = N-1$, we solve (4.18) exactly *(i.e., find h_N^x where the max is attained) by using duality arguments in 4.7.6 in the Appendix, without any uncertainty over-approximation. Therefore, the optimization problem (4.14) with constraint (4.18) is guaranteed to be feasible at $t = N-1$, with $u^\star_{N-1|t_\mathrm{f}}(\cdot)$ as a feasibility certificate. Let us denote the corresponding optimal policy from $t = N-1$ as:*

$$u_t^{\mathrm{MPC}}(x_t) = \bar{u}^\star_{t|t}. \tag{4.19}$$

Let policy (4.19) be applied to (2.10) in closed-loop, so that the system reaches the terminal set $\mathcal{X}_N$ at timestep $t+1$. Consider solving (4.18) at this step with a horizon length of $N_{t+1} = 1$. As, constraint (4.18) uses the same representation of the system uncertainty in satisfying (3.1e)-(3.1f) robustly as done in (3.5), we can infer that a candidate policy at timestep $(t+1)$ is

$$u_{t+1|t+1}(x_{t+1}) = Kx_{t+1}, \tag{4.20}$$

which is a feasible solution to the robust optimization problem (4.14) under constraint (4.18). Thus, (4.14) is guaranteed to remain feasible at $(t+1)$ with $N_{t+1} = 1$. This completes the proof. ■

Theorem 4.2 *Let Assumptions 3.1-3.4 in Chapter 3 hold and let the optimization problem (4.14) be feasible at timestep $t = 0$ with $N_t = N$. Then, $x_t \in \mathcal{X}_N$ for all $t \geq N$ and the origin of the closed-loop system:*

$$x_{t+1} = Ax_t + Bu^\star_{t|t}(x_t) + w_t, \ \forall t \geq 0. \tag{4.21}$$

obtained with (2.10)-(4.15) is ISS.

Proof *First, we have from Case-2 in the proof of Theorem 4.1 that at timestep $t = N-1$, the problem (4.14) is feasible with horizon $N_t = 1$ and therefore $x_t \in \mathcal{X}_N$ for all $t \geq N$.*

Now, consider the case of $t \geq N$, i.e., $N_t = 1$. Since (4.14) for $N_t = 1$ can be reformulated into a parametric QP, $V^{\text{MPC}}_{t\to t+1}(x_t, 0, 1)$ is continuous and piecewise quadratic in $\mathcal{X}_N$ with $V^{\text{MPC}}_{t\to t+1}(0, 0, 1) = 0$ [90]. Hence, under Assumptions 3.3-3.4, using the standard proof of [55, Theorem 23], we conclude that the origin of closed-loop system (4.21) is ISS according to Definition 3.2 in Chapter 3, and $V^{\text{MPC}}_{t\to t+1}(x_t, 0, 1)$ is ISS Lyapunov function for all $t \geq N$. ■

4.4 The ROA and Its Inner Approximation

We define the Region of Attraction (ROA) for Algorithm 1, denoted by $\mathcal{R}$, as the N-Step Robust Controllable Set to the terminal set $\mathcal{X}_N$ under the policy (4.15) for $t = 0$. This ensures that from Theorem 4.1 and Theorem 4.2 we have $\forall w_t \in \mathbb{W}$:

$$x_0 \in \mathcal{R} \implies \begin{cases} x_t \in \mathcal{X}, \ \forall t \geq 0, \ \text{and} \\ x_t \in \mathcal{X}_N \subseteq \mathcal{X}, \ \forall t \geq N, \end{cases}$$

where $x_{t+1} = Ax_t + Bu^{\text{MPC}}_t(x_t) + w_t$ for all $t \geq 0$. Thus, all the initial states in the ROA are steered to the terminal set $\mathcal{X}_N$ in maximum of N-steps while robustly satisfying (3.1e), where the origin of (4.21) is ISS. The ROA can be computed by solving problem (4.14) as a parametric optimization problem, with parameter x_t [6]. However, this computation may be prohibitive. We therefore use the fact that the ROA is convex and obtain its inner approximation using a set of vectors. Along each vector, we find an initial state for which (4.14) is feasible and which minimizes the inner product with the vector. The ROA is then approximated as the convex hull of these states. This is elaborated below.

Given a vector $v \in \mathbb{R}^n$, we define the following optimization problem at timestep $t = 0$:

$$\begin{aligned} P(N, v) = \\ \min_{\substack{x_0, \mathbf{M}_0^{(N)}, \bar{\mathbf{u}}_0^{(N)} \\ \bar{\mathbf{x}}_0^N}} \quad & v^\top x_0 \\ \text{s.t.}, \quad & (v^\perp)^\top x_0 = 0, \\ & G^{(N)}_{\text{eq}} \begin{bmatrix} (\bar{\mathbf{x}}_0^{(N)})^\top & \bar{x}_{N|0}^\top & (\bar{\mathbf{u}}_0^{(N)})^\top \end{bmatrix}^\top = b^{(N)}_{\text{eq}} x_0, \\ & \bar{x}_{0|0} = x_0, \\ & (4.9), (4.13), \ (\text{with } N_0 = N), \end{aligned} \tag{4.22}$$

with $\mathbf{f}^x_{\text{tight}}$ chosen as per (4.10), where $v^\perp \in \mathbb{R}^n$ is a vector perpendicular to $v \in \mathbb{R}^n$. Therefore, given a user-defined set of vectors $\mathcal{V} = \{v^{(1)}, v^{(2)}, \ldots, v^{(n_v)}\}$, problem (4.22) can be solved repeatedly and the convex hull of the optimal initial states $x_0^\star$ is an inner approximation to the ROA. It is clear from Algorithm 2 that the ROA approximation can improve, as the number of vectors in $\mathcal{V}$ increases.

Algorithm 2 Computation of the Approximate ROA

Inputs: Vectors $\mathcal{V} = \{v^{(1)}, v^{(2)}, \ldots, v^{(n_v)}\}$ and N
Initialize: $\mathcal{R}_{\text{ap}} = \varnothing$
for $v^{(i)} \in \mathcal{V}$ **do**
  Solve $P(N, v^{(i)})$ from (4.22). Let $x_0^\star$ be the optimal initial state from $P(N, v^{(i)})$.
  Set $\mathcal{R}_{\text{ap}} = \text{conv}\{\mathcal{R}_{\text{ap}} \cup \{x_0^\star\}\}$.
end for
Output: Approximate ROA: $\mathcal{R}_{\text{ap}} \subseteq \mathcal{R}$.

4.5 Numerical Simulations

We present our numerical simulations in this section. Algorithm 1 is implemented with $N = 3$ and N_t chosen as per (4.8) for all $t \geq 0$. We compare the performance of our Algorithm 1 with the polytopic tube MPCs of [15, Section 5] and [19, Section 3-4]. For our comparisons, we compute MPC solutions to (2.54) and utilize Remark 4.1. Gain K for constructing the terminal set $\mathcal{X}_N$ is chosen as $K = -[0.452, 0.418]$. The source code is available at https://github.com /monimoyb/RMPC_MixedUncertainty.

4.5.1 Comparison with [15]

For this comparison, we choose a horizon of 5 for the tube MPC method in [15, Section 5]. The tube cross section parameter (Z) is chosen as the minimal robust positive invariant set for system (2.10) under a feedback $u = -[1.2604, 0.7036]x$, and the terminal set ($\mathcal{X}_f$) is chosen as our terminal set $\mathcal{X}_N$ constructed with (3.5) and the aforementioned gain K.

ROA Comparison

Recall the notion of the ROA of Algorithm 1 from Section 4.4 and also its inner approximation obtained from Algorithm 2. We now choose a set of $N_{\text{init}} = 100$ initial states x_S, created by a 10×10 uniformly spaced grid of the set of state constraints in (2.54). From each of these initial state samples we check the feasibility of the tube MPC control problem in [15, Section 5]. The convex hull of the feasible initial state samples, which inner approximates its ROA, is then compared to the approximate ROA of Algorithm 1. This comparison is shown in Fig. 4.1. The approximate ROA of Algorithm 1 is about 1.04x in volume of that of the tube MPC in [15, Section 5].

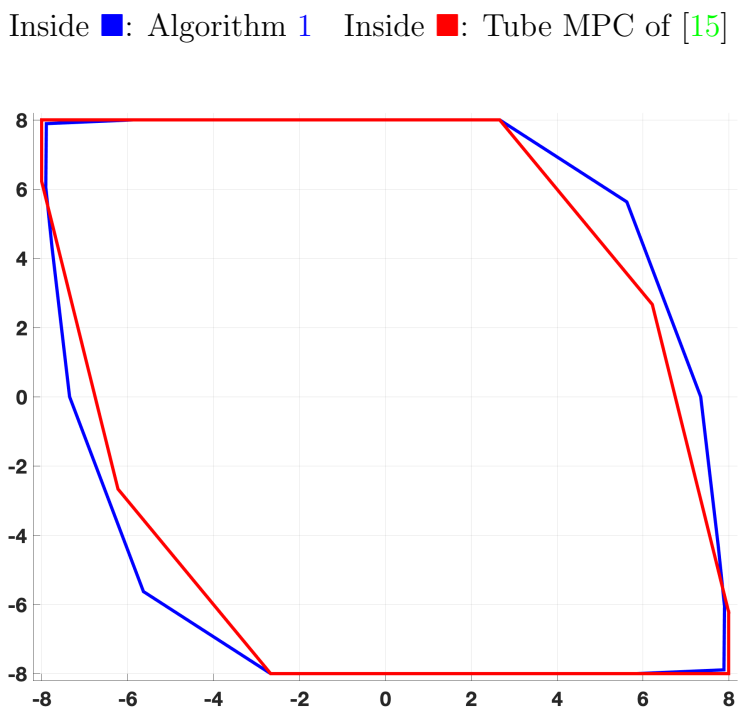


Figure 4.1: Comparison of the approximate ROA of Algorithm 1 with $N = 3$ and the approximate ROA of the tube MPC in [15, Section 5].

Computation Time Comparison

The advantage of our proposed approach becomes clearer upon comparing the online computation times. We see from Table 4.1 that for all relevant horizon lengths $N_t \in \{1, 2, 3\}$,

Table 4.1: Average computation times [sec]. Values are obtained with a MacBook Pro 16inch, 2019, 2.3 GHz 8-Core Intel Core i9, 16 GB memory, using the Gurobi solver [91].

Horizon	**Algorithm 1**		**Tube MPC in [15]**
	online	**offline**	**online**
$N_t = 1$	0.0019	0	0.0054
$N_t = 2$	0.0058	0.0279	0.1042
$N_t = 3$	0.0111	0.0687	0.2057

solving (4.14) is cheaper than computing the tube MPC online, even after adding the offline computation times required for bounds (4.27)-(4.31). The increase in computation times for such a polytopic tube MPC can be explained by an increase in the number of constraints imposed in the MPC problem with horizon N, as explained previously in Section 2.9.2.

Remark 4.2 *In the considered example, computing the set Z for the tube MPC in [15] required about 49 seconds offline. However, we have chosen not to include this in the comparison in Table 4.1, as any alternative simpler choice of Z is also valid. The choice of Z affects the size of the corresponding ROA [26].*

4.5.2 Comparison with [19]

Following the comparison of Chapter 3, we now compare our proposed approach with the polytopic tube MPC in [19] in this section.

ROA Comparison

Recall Remark 2.7. As a consequence, we compare the ellipsoidal ROA from in [19, Section 3] as used in Fig. 2.5 with the approximate ROA of Algorithm 1. This comparison is shown in Fig. 4.2. The approximate ROA of Algorithm 1 is about 30% larger in volume of that of the

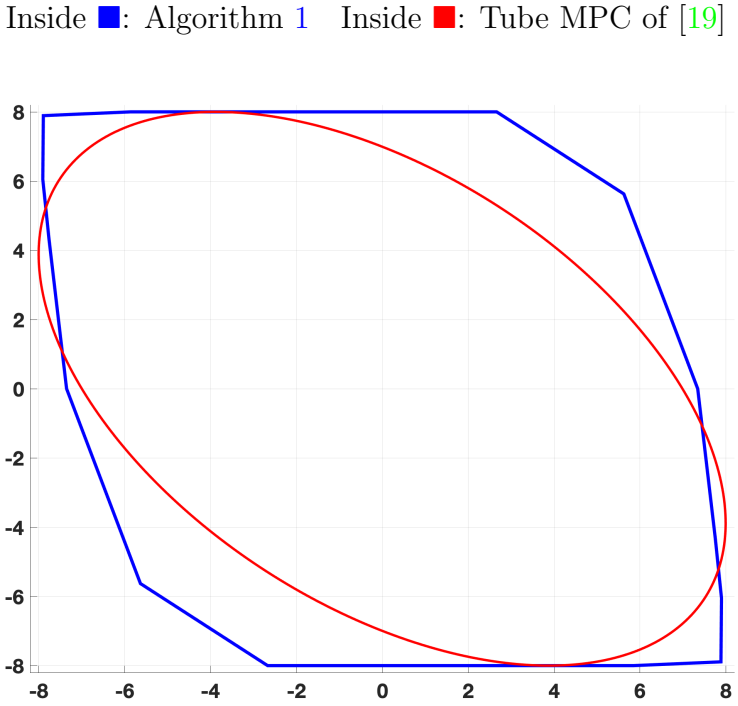


Figure 4.2: Comparison of the approximate ROA of Algorithm 1 with $N = 3$, and the ellipsoidal ROA of [19, Section 3].

polytopic tube MPC in [19, Section 3].

Computation Time Comparison

In this section we compare the computation times of our approach with those from [19, Section 3]. Table 4.2 shows the average computation times of Algorithm 1, both online and

Table 4.2: Comparison of average computation times [sec].

Horizon	**Algorithm 1**		**Tube MPC in [19]**	
	online	**offline**	**online**	**offline**
$N_t = 1$	0.0019	0	0.03	53.08
$N_t = 2$	0.0058	0.0279	0.03	53.08
$N_t = 3$	0.0111	0.0687	0.03	53.08

offline, for all relevant horizon lengths $N_t \in \{1, 2, 3\}$. Recall that due to Remark 2.7, the associated online times of [19] are independent of the horizon length N_t, as it only solves an LMI for obtaining the ellipsoidal ROA. We see from Table 4.2 that our proposed method in this chapter obtains lower online and offline computational times over the MPC of [19, Section 3]. Recall that the large offline computation time in [19, Section 3] is due to a logarithmic search required for optimizing a parameter (denoted by α in [19]). The offline computation times of our bounds (4.27)-(4.31) increase as we increase $N \geq 5$, however we limit to $N = 3$, since increasing the horizon length does not guarantee an enlargement of the ROA in our approach[3].

4.5.3 Comparison with the Robust MPC of Chapter 3

We point out that the approx. ROA of Algorithm 1 is about 4% larger in volume compared to the approx. ROA of the robust MPC proposed in Chapter 3. This comparison is shown in Fig. 4.3. Fig. 4.3 validates that for this considered example, the optimization-based constraint tightenings used in (4.14) lowers conservatism over using a net-additive uncertainty based worst-case bound (2.57).

However, for long horizons such as $N \geq 5$, computing bounds (4.27)-(4.31) can become computationally cumbersome. On the other hand, the alternative bounds (4.33)-(4.37) derived using binomial expansions can be rather conservative. In such a scenario, an approach such as the one proposed in Chapter 3 may be a more preferred option to utilize, which can yield an improved computation complexity vs conservatism trade-off over Algorithm 1. Furthermore, increasing the horizon N can enlarge the ROA of the robust MPC in Chapter 3, unlike Algorithm 1. Thus the choice of robust MPC algorithm depends on the specific problem at hand, and the computational resources available for control design, etc.

[3]This is because $\mathcal{R} \supseteq \mathcal{X}_N$ may not hold under the MPC policy (4.15), unlike typical MPC approaches.

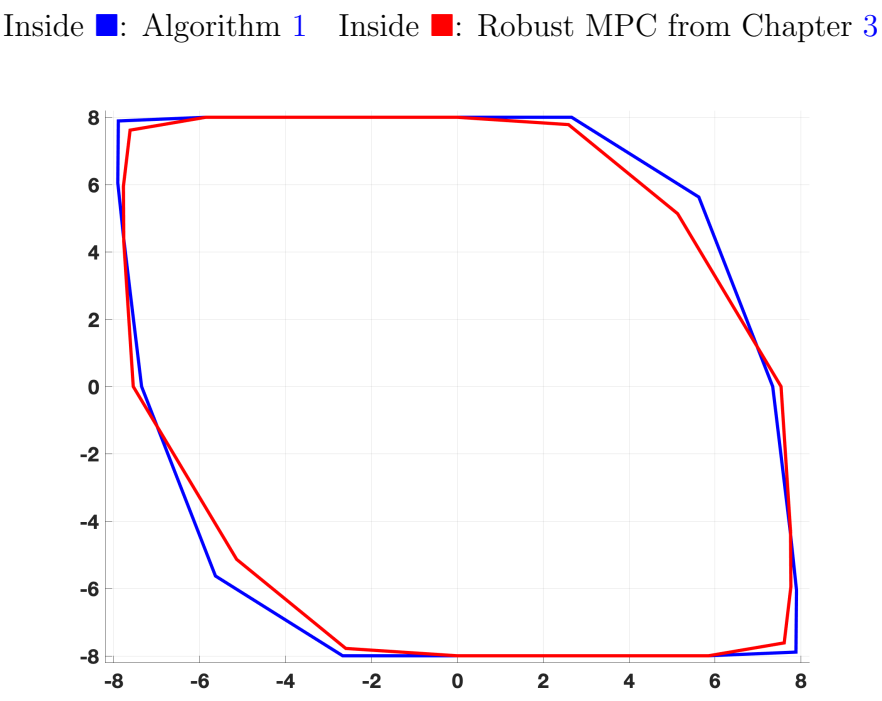


Figure 4.3: Comparison of the approximate ROA of Algorithm 1 and the robust MPC of Chapter 3 with $N = 3$.

4.6 Chapter Summary

We proposed another novel approach to design a robust MPC for constrained uncertain linear parameter varying systems. The uncertainty considered included both mismatch in the system dynamics matrices, and an additive disturbance. With set based bounds for each component of the model uncertainty being known at the time of control design, we proposed an optimization-based constraint tightening strategy utilizing these bounds. The proposed MPC achieved robust satisfaction of the imposed state and input constraints for all realizations of the model uncertainty. We further proved Input to State Stability of the origin. With numerical simulations, we demonstrated that our controller obtained at least 3x and up to 20x speedup in online control computations and an approximately 4% larger ROA by volume, compared to the tube MPC in [15]. We obtained up to 3x speedup in online control computations and an approximately 30% larger ROA by volume compared to the state-of-the-art polytopic tube MPC of [19]. Due to the proposed optimization-based constraint tightening approach in this chapter, we also obtained a 4% decrease in conservatism over the robust MPC proposed in Chapter 3.

4.7 Appendix

4.7.1 Matrix Definitions

The prediction dynamics matrices $\mathbf{A}^x, \mathbf{A}^u, \mathbf{A}^{\Delta u}$ and $\mathbf{A}^w$ in (4.3) for a horizon length[4] of $\bar{N}$ are given by

$$\mathbf{A}^x = \begin{bmatrix} A_\Delta & 0 & 0 & \dots & 0 \\ A_\Delta\Delta_A & A_\Delta & 0 & \dots & 0 \\ A_\Delta^2\Delta_A & A_\Delta\Delta_A & A_\Delta & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_\Delta^{\bar{N}-1}\Delta_A & A_\Delta^{\bar{N}-2}\Delta_A & \dots & \dots & A_\Delta \end{bmatrix} \in \mathbb{R}^{n\bar{N}\times n\bar{N}},$$

$$\mathbf{A}^u = \begin{bmatrix} B_\Delta & 0 & 0 & \dots & 0 \\ A_\Delta\Delta_B & B_\Delta & 0 & \dots & 0 \\ A_\Delta^2\Delta_B & A_\Delta\Delta_B & B_\Delta & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_\Delta^{\bar{N}-1}\Delta_B & A_\Delta^{\bar{N}-2}\Delta_B & \dots & \dots & B_\Delta \end{bmatrix} \in \mathbb{R}^{n\bar{N}\times m\bar{N}},$$

$$\mathbf{A}^{\Delta u} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ A_\Delta\bar{B} & 0 & 0 & \dots & 0 \\ A_\Delta^2\bar{B} & A_\Delta\bar{B} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_\Delta^{\bar{N}-1}\bar{B} & A_\Delta^{\bar{N}-2}\bar{B} & \dots & A_\Delta\bar{B} & 0 \end{bmatrix} \in \mathbb{R}^{n\bar{N}\times m\bar{N}},$$

$$\mathbf{A}^w = \begin{bmatrix} I_n & 0 & 0 & \dots & 0 \\ A_\Delta & I_n & 0 & \dots & 0 \\ A_\Delta^2 & A_\Delta & I_n & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_\Delta^{\bar{N}-1} & A_\Delta^{\bar{N}-2} & \dots & \dots & I_n \end{bmatrix} \in \mathbb{R}^{n\bar{N}\times n\bar{N}},$$

where $A_\Delta = (\bar{A} + \Delta_A) \in \mathcal{P}_{A_\Delta}$ and $B_\Delta = (\bar{B} + \Delta_B) \in \mathcal{P}_{B_\Delta}$. We write matrices $\bar{\mathbf{A}}_1$ and $\mathbf{A}_\delta \in \mathbb{R}^{n\bar{N}\times n\bar{N}}$ as:

$$\bar{\mathbf{A}}_1 = \begin{bmatrix} I_n & 0 & 0 & \dots & 0 \\ \bar{A} & I_n & 0 & \dots & 0 \\ \bar{A}^2 & \bar{A} & I_n & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{A}^{\bar{N}-1} & \bar{A}^{\bar{N}-2} & \dots & \dots & I_n \end{bmatrix}, \quad \mathbf{A}_\delta = (\mathbf{A}^w - \bar{\mathbf{A}}_1),$$

[4]Equation (4.3) was introduced with a fixed horizon length of N, i.e., $\bar{N} \leftarrow N$. However, dimensions of these matrices vary as horizon length is varied later in Section 4.2.3.

which gives $\mathbf{A}^x = \bar{\mathbf{A}} + \Big(\bar{\mathbf{A}}_1 + \mathbf{A}_\delta\Big)\mathbf{\Delta}_A$, $\mathbf{A}^u = \bar{\mathbf{B}} + \Big(\bar{\mathbf{A}}_1 + \mathbf{A}_\delta\Big)\mathbf{\Delta}_B$, and $\mathbf{A}^{\Delta u} = \Big(\bar{\mathbf{A}}_1 - \mathbf{I}_n + \mathbf{A}_\delta\Big)\bar{\mathbf{B}}$. The matrix $\bar{\mathbf{A}}_v$ is written as $\bar{\mathbf{A}}_v = \begin{bmatrix} A_v^{(1)} & A_v^{(2)} & \dots & A_v^{(\bar{N}-1)} \end{bmatrix}$, where matrices $\{A_v^{(1)}, A_v^{(2)}, \dots, A_v^{(\bar{N}-1)}\}$ are given as

$$A_v^{(1)} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ I_n & 0 & 0 & \dots & 0 \\ 0 & I_n & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & I_n & 0 \end{bmatrix}, \; A_v^{(2)} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ I_n & 0 & 0 & \dots & 0 \\ 0 & I_n & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & I_n & \dots & 0 \end{bmatrix},$$

and analogously for $A_v^{(3)}, A_v^{(4)}, \dots, A_v^{(\bar{N}-1)}$.

This gives $\mathbf{A}^w = \mathbf{I}_n + \bar{\mathbf{A}}_v \mathbf{A}_\Delta$, with $\mathbf{I}_n = (I_{\bar{N}} \otimes I_n)$, and

$$\mathbf{A}_\Delta = \begin{bmatrix} I_{\bar{N}} \otimes A_\Delta \\ I_{\bar{N}} \otimes A_\Delta^2 \\ \vdots \\ I_{\bar{N}} \otimes A_\Delta^{\bar{N}-1} \end{bmatrix} \in \mathbb{R}^{n\bar{N}(\bar{N}-1) \times n\bar{N}}. \tag{4.23}$$

4.7.2 Deriving (4.7) from (4.6)

Using (4.5) in (4.3), constraints (4.6) can be written as:

$$\begin{aligned} \mathbf{F}^x \Bigg(& \bar{\mathbf{A}}\bar{\mathbf{x}}_t + \bar{\mathbf{A}}_1 \mathbf{\Delta}_A \bar{\mathbf{x}}_t + (\mathbf{A}_\delta \mathbf{\Delta}_A)\bar{\mathbf{x}}_t + \bar{\mathbf{B}}\mathbf{u}_t + \bar{\mathbf{A}}_1 \mathbf{\Delta}_B \mathbf{u}_t + \\ & \cdots + (\mathbf{A}_\delta \mathbf{\Delta}_B)\mathbf{u}_t + (\bar{\mathbf{A}}_1 - \mathbf{I}_n + \mathbf{A}_\delta)\bar{\mathbf{B}}\Delta\mathbf{u}_t + \mathbf{w}_t + \bar{\mathbf{A}}_v \mathbf{A}_\Delta \mathbf{w}_t \Bigg) \leq \mathbf{f}^x, \end{aligned} \tag{4.24}$$

$\forall \Delta_A \in \mathcal{P}_A, \; \forall \Delta_B \in \mathcal{P}_B, \; \forall w_t \in \mathbb{W}.$

We obtain an upper bound for the left hand side of inequality (4.24) row-wise as follows:

$$\begin{aligned} & \mathbf{F}_i^x(\bar{\mathbf{A}}\bar{\mathbf{x}}_t + \bar{\mathbf{B}}\mathbf{u}_t + (\bar{\mathbf{A}}_1 - \mathbf{I}_n)\bar{\mathbf{B}}\Delta\mathbf{u}_t + \mathbf{w}_t) + \mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_A \bar{\mathbf{x}}_t + \mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_B \mathbf{u}_t + \mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_A \bar{\mathbf{x}}_t + \\ & \qquad \cdots + \mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_B \mathbf{u}_t + \mathbf{F}_i^x \mathbf{A}_\delta \bar{\mathbf{B}}\Delta\mathbf{u}_t + \mathbf{F}_i^x \bar{\mathbf{A}}_v \mathbf{A}_\Delta \mathbf{w}_t, \\ & \leq \mathbf{F}_i^x(\bar{\mathbf{A}}\bar{\mathbf{x}}_t + \bar{\mathbf{B}}\mathbf{u}_t + (\bar{\mathbf{A}}_1 - \mathbf{I}_n)\bar{\mathbf{B}}\Delta\mathbf{u}_t + \mathbf{w}_t) + \mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_A \bar{\mathbf{x}}_t + \mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_B \mathbf{u}_t + \|\mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_A\|_* \|\bar{\mathbf{x}}_t\| + \\ & \qquad \cdots + \|\mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_B\|_* \|\mathbf{u}_t\| + \|\mathbf{F}_i^x \mathbf{A}_\delta \bar{\mathbf{B}}\|_* \|\Delta\mathbf{u}_t\| + \|\mathbf{F}_i^x \bar{\mathbf{A}}_v \mathbf{A}_\Delta\|_* \|\mathbf{w}_t\|, \end{aligned} \tag{4.25}$$

for rows $i \in \{1, 2, \dots, r(\bar{N}-1) + r_N\}$, where we have used the Hölder's inequality. Using bounds (4.27)-(4.31) in (4.25) then yields (4.7). Note that in (4.7) $\bar{N} \leftarrow N$.

4.7.3 Bounding Nominal Trajectory Perturbations

For any horizon length[5] of $\bar{N} \in \{2, 3, \dots, N\}$, we first bound:

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta\|_*, \text{ where using (4.4) we have } \mathbf{A}_\delta = \bar{\mathbf{A}}_v \begin{bmatrix} I_{\bar{N}} \otimes (A_\Delta - \bar{A}) \\ I_{\bar{N}} \otimes (A_\Delta^2 - \bar{A}^2) \\ \vdots \\ I_{\bar{N}} \otimes (A_\Delta^{\bar{N}-1} - \bar{A}^{\bar{N}-1}) \end{bmatrix}. \tag{4.26}$$

Note that for all $A_\Delta \in \mathcal{P}_{A_\Delta} \implies A_\Delta^n \in \mathcal{P}_{A_\Delta}^n$, for $n \in \{1, 2, \dots, \bar{N}-1\}$, where $\mathcal{P}_{A_\Delta}^n$ is the set of all matrices that can be written as a convex combination of matrices obtained with the product of *all possible combinations* of n matrices out of $\{(\bar{A} + \Delta_A^{(1)}), (\bar{A} + \Delta_A^{(2)}), \dots, (\bar{A} + \Delta_A^{(n_a)})\}$. Hence

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta\|_* \leq \max_{\substack{\Delta_1 \in \mathcal{P}_{A_\Delta} \\ \Delta_2 \in \mathcal{P}_{A_\Delta}^2 \\ \vdots \\ \Delta_{\bar{N}-1} \in \mathcal{P}_{A_\Delta}^{\bar{N}-1}}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v \begin{bmatrix} I_N \otimes (\Delta_1 - \bar{A}) \\ I_N \otimes (\Delta_2 - \bar{A}^2) \\ \vdots \\ I_N \otimes (\Delta_{\bar{N}-1} - \bar{A}^{\bar{N}-1}) \end{bmatrix} \|_* = \mathbf{t}_0^i, \tag{4.27}$$

where we have relaxed all the equality constraints among the matrices $\{\Delta_1, \Delta_2, \dots, \Delta_{\bar{N}-1}\}$. Using the above bound (4.27), we get

$$\max_{\substack{A_\Delta \in \mathcal{P}_{A_\Delta} \\ \Delta_A \in \mathcal{P}_A}} \|\mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_A\|_* \leq \mathbf{t}_0^i \max_{\Delta_A \in \mathcal{P}_A} \|\mathbf{\Delta}_A\|_p = \mathbf{t}_1^i, \tag{4.28}$$

where we have used the consistency property of induced norms, for any $p = 1, 2, \infty$. Similarly, bounding terms

$$\max_{\substack{A_\Delta \in \mathcal{P}_{A_\Delta} \\ \Delta_B \in \mathcal{P}_B}} \|\mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_B\|_* \leq \mathbf{t}_0^i \max_{\Delta_B \in \mathcal{P}_B} \|\mathbf{\Delta}_B\|_p = \mathbf{t}_2^i, \tag{4.29}$$

and

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta \bar{\mathbf{B}}\|_* \leq \mathbf{t}_0^i \|\bar{\mathbf{B}}\|_p = \mathbf{t}_3^i, \tag{4.30}$$

and finally

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v \mathbf{A}_\Delta\|_* \leq \max_{\substack{\Delta_1 \in \mathcal{P}_{A_\Delta} \\ \Delta_2 \in \mathcal{P}_{A_\Delta}^2 \\ \vdots \\ \Delta_{\bar{N}-1} \in \mathcal{P}_{A_\Delta}^{\bar{N}-1}}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v \begin{bmatrix} I_{\bar{N}} \otimes \Delta_1 \\ I_{\bar{N}} \otimes \Delta_2 \\ \vdots \\ I_{\bar{N}} \otimes \Delta_{\bar{N}-1} \end{bmatrix} \|_* = \mathbf{t}_w^i, \tag{4.31}$$

[5]Note, also the bounds in Section 4.2.2 were introduced with a fixed horizon length of N, i.e., $\bar{N} \leftarrow N$.

for $i \in \{1, 2, \ldots, r(\bar{N}-1)+r_N\}$. Problems (4.27)-(4.31) are maximizing convex functions of the decision variables over convex and compact domains. Therefore, these maximum bounds are attained at the extreme points, i.e., vertices of the convex sets $\{\mathcal{P}_{A_\Delta}, \mathcal{P}_{A_\Delta}^2, \ldots, \mathcal{P}_{A_\Delta}^{\bar{N}-1}\}$, $\mathcal{P}_A$ and $\mathcal{P}_B$. Consequently, the optimal values of (4.27)-(4.31) can be obtained by evaluating the values of each of the terms in (4.27)-(4.31) at all possible combinations of such extreme points. Since such a vertex enumeration strategy scales poorly with the horizon length N, a computationally cheaper alternative to bounds (4.27)-(4.31) is presented next.

4.7.4 Computationally Efficient Alternatives of Bounds (4.27)-(4.31)

Recall the optimization problem from (4.27), given by

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta\|_*, \text{ with } \mathbf{A}_\delta \text{ from (4.26)}. \tag{4.32}$$

Using the triangle and Hölder's inequalities, and the submultiplicativity and consistency properties of induced norms, (4.32) can be upper bounded for any cut-off horizon $\tilde{N} < \bar{N}$ as follows:

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta\|_* \leq \tilde{\mathbf{t}}_0^i + \hat{\mathbf{t}}_0^i = \mathbf{t}_0^i, \tag{4.33}$$

with

$$\tilde{\mathbf{t}}_0^i = \max_{\substack{\Delta_1 \in \mathcal{P}_{A_\Delta} \\ \vdots \\ \Delta_{\tilde{N}-1} \in \mathcal{P}_{A_\Delta}^{\tilde{N}-1}}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v^{1:(\tilde{N}-1)} \begin{bmatrix} I_{\bar{N}} \otimes (\Delta_1 - \bar{A}) \\ I_{\bar{N}} \otimes (\Delta_2 - \bar{A}^2) \\ \vdots \\ I_{\bar{N}} \otimes (\Delta_{\tilde{N}-1} - \bar{A}^{\tilde{N}-1}) \end{bmatrix} \|_*$$

where $\bar{\mathbf{A}}_v^{n_1:n_2}$ denotes $\begin{bmatrix} A_v^{(n_1)} & A_v^{(n_1+1)} & \ldots & A_v^{(n_2)} \end{bmatrix}$, with the associated matrices defined in Appendix 4.7.1, $\mathbf{F}_i^x[n_1 : n_2]$ denotes the n_1 to n_2 columns of the row vector $\mathbf{F}_i^x$, for $i \in \{1, 2, \ldots, r(\bar{N}-1)+r_N\}$, and

$$\hat{\mathbf{t}}_0^i = \max_{\Delta_A \in \mathcal{P}_A} \left(\sum_{j=\tilde{N}+1}^{\bar{N}} \|\mathbf{F}_i^x[(j-1)n+1 : jn]\|_* \Big(\sum_{k=1}^{j-\tilde{N}} \Big(\sum_{l=1}^{j-k} \binom{j-k}{l} \|\bar{A}\|_p^{j-k-l} \|\Delta_A\|_p^l \Big) \Big) \right).$$

Using the above derived bound (4.33) we obtain:

$$\max_{\substack{A_\Delta \in \mathcal{P}_{A_\Delta} \\ \Delta_A \in \mathcal{P}_A}} \|\mathbf{F}_i^x \mathbf{A}_\delta \boldsymbol{\Delta}_A\|_* \leq \mathbf{t}_0^i \max_{\Delta_A \in \mathcal{P}_A} \|\boldsymbol{\Delta}_A\|_p = \mathbf{t}_1^i, \tag{4.34}$$

where we have used the consistency property of induced norms, for any $p = 1, 2, \infty$. Similarly, we bound

$$\max_{\substack{A_\Delta \in \mathcal{P}_{A_\Delta} \\ \Delta_B \in \mathcal{P}_B}} \|\mathbf{F}_i^x \mathbf{A}_\delta \mathbf{\Delta}_B\|_* \leq \mathbf{t}_0^i \max_{\Delta_B \in \mathcal{P}_B} \|\mathbf{\Delta}_B\|_p = \mathbf{t}_2^i, \tag{4.35}$$

and,

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \mathbf{A}_\delta \bar{\mathbf{B}}\|_* \leq \mathbf{t}_0^i \|\bar{\mathbf{B}}\|_p = \mathbf{t}_3^i, \tag{4.36}$$

and finally using $\mathbf{A}_\Delta$ from (4.23)

$$\max_{A_\Delta \in \mathcal{P}_{A_\Delta}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v \mathbf{A}_\Delta\|_* \leq \tilde{\mathbf{t}}_w^i + \hat{\mathbf{t}}_w^i = \mathbf{t}_w^i, \tag{4.37}$$

for all $i \in \{1, 2, \ldots, r(\bar{N}-1) + r_N\}$, where

$$\tilde{\mathbf{t}}_w^i = \max_{\substack{\Delta_1 \in \mathcal{P}_{A_\Delta} \\ \vdots \\ \Delta_{\tilde{N}-1} \in \mathcal{P}_{A_\Delta}^{\tilde{N}-1}}} \|\mathbf{F}_i^x \bar{\mathbf{A}}_v^{1:(\tilde{N}-1)} \begin{bmatrix} I_{\bar{N}} \otimes \Delta_1 \\ I_{\bar{N}} \otimes \Delta_2 \\ \vdots \\ I_{\bar{N}} \otimes \Delta_{\tilde{N}-1} \end{bmatrix} \|_*,$$

and

$$\hat{\mathbf{t}}_w^i = \max_{\Delta_A \in \mathcal{P}_A} \left(\sum_{j=\tilde{N}}^{\bar{N}-1} \|\mathbf{F}_i^x A_v^{(j)}\|_* \Big(\|(I_{\bar{N}} \otimes \bar{A})^j\|_p + \sum_{k=1}^{j} \binom{j}{k} \|(I_{\bar{N}} \otimes \bar{A})\|_p^{j-k} \|(I_{\bar{N}} \otimes \Delta_A)\|_p^k \Big) \right),$$

where we have used the property of two matrices X and Y yielding:

$$\|(X+Y)^d\|_p \leq \|X^d\|_p + \sum_{k=1}^{d} \binom{d}{k} \|X\|_p^{d-k} \|Y\|_p^k,$$
$$\forall d \in \{\tilde{N}, \tilde{N}+1, \ldots, \bar{N}-1\}.$$

This cut-off horizon $\tilde{N}$ can be chosen based on the available computational resources at the expense of more conservatism over (4.27)-(4.31).

4.7.5 Obtaining (4.9) from (4.7)

Here we derive (4.9) from (4.7). Using bounds (4.11) and (4.27)-(4.31) and policy parametrization (3.3), constraints (4.7) can be satisfied by imposing:

$$\max_{\mathbf{w}_t \in \mathbf{W}} \Big(\mathbf{F}_i^x (\bar{\mathbf{A}} \bar{\mathbf{x}}_t^{(N_t)} + \bar{\mathbf{B}} (\mathbf{M}_t^{(N_t)} \mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)}) + (\bar{\mathbf{A}}_1 - \mathbf{I}_n) \bar{\mathbf{B}} \mathbf{M}_t^{(N_t)} \mathbf{w}_t + \mathbf{w}_t) + \mathbf{t}_{\delta 1}^{(N_t),i} \|\bar{\mathbf{x}}_t^{(N_t)}\| + \cdots$$
$$+ (\mathbf{t}_2^{(N_t),i} + \mathbf{t}_{\delta B}^{(N_t),i}) \|\mathbf{M}_t^{(N_t)} \mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)}\| + \mathbf{t}_3^{(N_t),i} \|\mathbf{M}_t^{(N_t)} \mathbf{w}_t\| + \mathbf{t}_w^{(N_t),i} \mathbf{w}_{\max} \Big) \leq \mathbf{f}_i^x, \tag{4.38}$$

where using (4.11) we have used the Hölder's and the triangle inequality to bound $\mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_A \bar{\mathbf{x}}_t^{(N_t)}$ and $\mathbf{F}_i^x \bar{\mathbf{A}}_1 \mathbf{\Delta}_B (\mathbf{M}_t^{(N_t)} \mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)})$ for all rows $i \in \{1, 2, \ldots, r(N_t - 1) + r_N\}$. Use the induced norm consistency property and the triangle inequality in (4.38) as:

$$\begin{aligned}
&(\mathbf{t}_2^{(N_t),i} + \mathbf{t}_{\delta B}^{(N_t),i}) \|\mathbf{M}_t^{(N_t)} \mathbf{w}_t + \bar{\mathbf{u}}_t^{(N_t)}\| + \mathbf{t}_3^{(N_t)} \|\mathbf{M}_t^{(N_t)} \mathbf{w}_t\|, \\
&\leq (\mathbf{t}_2^{(N_t),i} + \mathbf{t}_{\delta B}^{(N_t),i} + \mathbf{t}_3^{(N_t),i}) \|\mathbf{M}_t^{(N_t)}\|_p \mathbf{w}_{\max} + (\mathbf{t}_2^{(N_t),i} + \mathbf{t}_{\delta B}^{(N_t),i}) \|\bar{\mathbf{u}}_t^{(N_t)}\|, \\
&\leq \mathbf{t}_{\delta 3}^{(N_t),i} \|\mathbf{M}_t^{(N_t)}\|_p \mathbf{w}_{\max} + \mathbf{t}_{\delta 2}^{(N_t),i} \|\bar{\mathbf{u}}_t^{(N_t)}\|,
\end{aligned} \tag{4.39}$$

for any $p = 1, 2, \infty$, where we have used the definitions (4.11). Using (4.39) in (4.38) for all rows $i \in \{1, 2, \ldots, r(N_t - 1) + r_N\}$, we define

$$\mathbf{f}_{\text{tight}}^x = \mathbf{f}^x - \mathbf{t}_{\delta 1}^{(N_t)} \|\bar{\mathbf{x}}_t^{(N_t)}\| - \mathbf{t}_{\delta 3}^{(N_t)} \|\mathbf{M}_t^{(N_t)}\|_p \mathbf{w}_{\max} - \mathbf{t}_{\delta 2}^{(N_t)} \|\bar{\mathbf{u}}_t^{(N_t)}\| - \mathbf{t}_w^{(N_t)} \mathbf{w}_{\max},$$

which yields (4.9) with tightened constraints (4.10).

4.7.6 Reformulation of (4.14) via Duality of Convex Programs

We again consider the following two cases for satisfying the robust state constraints (4.9).

Case 1: ($N_t \geq 2$, i.e., $t \leq N - 2$) Constraints (4.9a) can be satisfied using duality of convex programs by solving:

$$\begin{aligned}
&\mathbf{F}^x (\bar{\mathbf{A}} \bar{\mathbf{x}}_t^{(N_t)} + \bar{\mathbf{B}} \bar{\mathbf{u}}_t^{(N_t)}) + \Lambda^{(N_t)} \mathbf{h}^w \leq \mathbf{f}_{\text{tight}}^x, \\
&\Lambda^{(N_t)} \geq 0, \\
&\Lambda^{(N_t)} \mathbf{H}^w = \Big(\mathbf{F}^x (\bar{\mathbf{B}} \mathbf{M}_t^{(N_t)} + (\bar{\mathbf{A}}_1 - \mathbf{I}_n) \bar{\mathbf{B}} \mathbf{M}_t^{(N_t)} + \mathbf{I}_n) \Big),
\end{aligned}$$

where $\mathbf{f}_{\text{tight}}^x$ is obtained from (4.10), and dual variables $\Lambda^{(N_t)} \in \mathbb{R}^{(r(N_t - 1) + r_N) \times a N_t}$.

Case 2: ($N_t = 1$, i.e., $t \geq N - 1$) Consider the case of $N_t = 1$. As pointed out in (4.9b), the robust state constraint for this case can be simplified and written as

$$\max_{\substack{w_t \in \mathbb{W} \\ \Delta_A \in \mathcal{P}_A \\ \Delta_B \in \mathcal{P}_B}} H_N^x ((\bar{A} + \Delta_A) \bar{\mathbf{x}}_t^{(1)} + (\bar{B} + \Delta_B) \bar{\mathbf{u}}_t^{(1)} + w_t) \leq h_N^x,$$

which we must solve *exactly* (i.e., find h_N^x where the max is attained) for the uncertainty representation $w_t \in \mathbb{W}$, $\Delta_A \in \mathcal{P}_A$ and $\Delta_B \in \mathcal{P}_B$, in order for guarantees of Theorem 4.1 to hold. Using duality of convex programs [67] one can write the robust state constraints (4.9b) equivalently as:

$$\begin{aligned}
&H_N^x ((\bar{A} + \Delta_A^{(j)}) \bar{\mathbf{x}}_t^{(1)} + (\bar{B} + \Delta_B^{(k)}) \bar{\mathbf{u}}_t^{(1)}) + \Lambda^{(1)} h^w \leq h_N^x, \\
&\Lambda^{(1)} \geq 0, \; H_N^x = \Lambda^{(1)} H^w, \\
&\forall j \in \{1, 2, \ldots, n_a\}, \; \forall k \in \{1, 2, \ldots, n_b\},
\end{aligned} \tag{4.40}$$

where dual variables $\Lambda^{(1)} \in \mathbb{R}^{r_N \times a}$.

Input Constraints: Considering the robust input constraints (4.13) for any $N_t \in \{1, 2, \ldots, N\}$, one can similarly show that this is equivalent to:

$$(\gamma^{(N_t)})^\top \mathbf{h}^w \leq \mathbf{h}^u - \mathbf{H}^u \bar{\mathbf{u}}_t^{(N_t)},$$
$$(\mathbf{H}^u \mathbf{M}_t^{(N_t)})^\top = (\mathbf{H}^w)^\top \gamma^{(N_t)},\ \gamma^{(N_t)} \geq 0,$$

by introducing decision variables of $\gamma^{(N_t)} \in \mathbb{R}^{aN_t \times oN_t}$ in (4.14) for each horizon length $N_t \in \{1, 2, \ldots, N\}$.

Chapter 5

Learning Non-Parametric Model Uncertainty in Robust MPC

This chapter is based on the published work [30]. We focus on the learning of model uncertainty in robust MPC in this chapter. The first case considered is the case of an additive non-parametric uncertainty, i.e., model (M3) in Chapter 2. The additive uncertainty is assumed globally Lipschitz, with a known Lipschitz constant.

5.1 Summary of Contributions

We utilize a non-parametric recursive system identification strategy [24], which identifies the *graph* of the uncertainty from data using its Lipschitz property. The identification is successively refined with recorded data. Our main contributions are:

- We provide set based bounds containing all possible realizations of the system uncertainty, using its Lipschitz property. This in contrast to the probabilistic nature of bounds in [7, 8, 9], due to the use of GP regression. Our uncertainty set bounds are modified successively with set intersections upon gathering new data.
- Utilizing the above bounds on system uncertainty, we synthesize a robust adaptive MPC controller by solving convex optimization problems, satisfying imposed state and input constraints. We prove its recursive feasibility, extending feasibility guarantees of [22, 92, 31] in presence of state dependent uncertainty. We further demonstrate the validity and efficacy of the proposed approach through a detailed numerical simulation.

5.2 Problem Formulation

The system model is given by model (M3) introduced in Chapter 2, i.e.,

$$x_{t+1} = Ax_t + Bu_t + d(x_t), \tag{5.1}$$

where $d(x_t)$ constitutes the system uncertainty, which is L_d-Lipschitz in its convex and closed domain $\text{dom}(d)$ with a known L_d. The system dynamics are subject to polytopic state and input constraints of the form:

$$\mathcal{X} = \{x \in \mathbb{R}^n \mid H_x x \leq h_x\}, \tag{5.2a}$$
$$\mathcal{U} = \{u \in \mathbb{R}^m \mid H_u u \leq h_u\}, \tag{5.2b}$$

where we assume $\mathcal{X} \subseteq \text{dom}(d)$.

5.2.1 Robust MPC Problem

We wish to design a robust MPC controller by finding solutions to the following optimization problem at each timestep t:

$$\begin{aligned}
\min_{u_{t|t}, u_{t+1|t}(\cdot), \ldots, u_{t+N-1|t}(\cdot)} \quad & \sum_{k=t}^{t+N-1} (\bar{x}_{k|t}^\top Q \bar{x}_{k|t} + u_{k|t}^\top(\bar{x}_{k|t}) R u_{k|t}(\bar{x}_{k|t})) + \bar{x}_{t+N|t}^\top P_N \bar{x}_{t+N|t} \\
\text{s.t.,} \quad & x_{k+1|t} = A x_{k|t} + B u_{k|t}(x_{k|t}) + d(x_{k|t}), \\
& \bar{x}_{k+1|t} = A \bar{x}_{k|t} + B u_{k|t}(\bar{x}_{k|t}) + \bar{d}(\bar{x}_{k|t}), \\
& H_x x_{k+1|t} \leq h_x, \\
& H_u u_{k|t}(x_{k|t}) \leq h_u, \\
& \forall d(x_{k|t}) \in \mathcal{D}(x_{k|t}), \\
& \forall k = \{t, t+1, \ldots t+N-1\}, \\
& x_{t+N|t} \in \mathcal{X}_N, \\
& x_{t|t} = \bar{x}_{t|t} = x_t,
\end{aligned} \tag{5.3}$$

where $x_{k|t}$ is the predicted state after applying the predicted policy $\{u_{t|t}, \ldots, u_{k-1|t}(x_{k-1|t})\}$ for $k = \{t+1, \ldots, t+N\}$ to system (5.1), $\mathcal{X}_N$ is the terminal set, $P_N \succ 0$ is the terminal cost and matrices $Q, R \succ 0$ are weight matrices.

5.2.2 Control Policy Parametrization

We restrict ourselves to the affine disturbance feedback parametrization, as per (P2) in Chapter 2. For all $k \in \{t, \cdots, t+N-1\}$ over the MPC horizon (of length N), the control policy is given as:

$$u_{k|t}(x_{k|t}) = \sum_{l=t}^{k-1} M_{k,l|t} d(x_{l|t}) + v_{k|t}, \tag{5.4}$$

where $M_{k|t}$ are the *planned* feedback gains at timestep t and $v_{k|t}$ are the auxiliary inputs. Let us define $\mathbf{d}(x_t) = [d(x_{t|t}), \cdots, d(x_{t+N-1|t})]^\top \in \mathbb{R}^{nN}$. Then the sequence of predicted inputs from (5.4) can be compactly written as $\mathbf{u}_t = \mathbf{M}_t \mathbf{d}(x_t) + \mathbf{v}_t$ at any timestep t, where $\mathbf{M}_t \in \mathbb{R}^{mN \times nN}$ and $\mathbf{v}_t \in \mathbb{R}^{mN}$ are shown in (2.28).

5.3 Uncertainty Learning and Adaptation

At every time instant t, we assume that we have access to measurements $d(x_i)$ for all $i = \{0, 1, \ldots, t-1\}$, that is, the realizations of the uncertainty function.

5.3.1 Successive Graph Approximation

Definition 5.1 (Graph) *The graph of a function $f : \mathbb{R}^n \to \mathbb{R}^n$ is defined as the set*

$$G(f) = \{(x, f(x)) \in \mathbb{R}^n \times \mathbb{R}^n | \ \forall x \in \mathrm{dom}(f)\}.$$

We use quadratic constraints (QCs) as our main tool to approximate the graph of a function. A definition appropriate for our purposes is presented below.

Definition 5.2 (QC Satisfaction) *A set $\mathcal{A} \subset \mathbb{R}^{2n}$ is said to satisfy the quadratic constraint specified by symmetric matrix Q_c if*

$$\begin{bmatrix} x \\ 1 \end{bmatrix}^\top Q_c \begin{bmatrix} x \\ 1 \end{bmatrix} \leq 0, \quad \forall x \in \mathcal{A}.$$

The following proposition uses a QC to characterise a coarse approximation of the graph of an L_d-Lipschitz function.

Proposition 5.1 *The graph $G(d)$ of the L_d-Lipschitz function $d(\cdot)$ inferred at any timestep t, using the measurement $(x_i, d(x_i))$ for any $0 \leq i < t$, satisfies the QC specified by the matrix*

$$Q_L^d(x_i) = \begin{bmatrix} -L_d^2 I_n & \mathbf{0}_{n\times n} & L_d^2 x_i \\ \mathbf{0}_{n\times n} & I_n & -d(x_i) \\ L_d^2(x_i)^\top & -d^\top(x_i) & -L_d^2(x_i)^\top x_i + d^\top(x_i) d(x_i) \end{bmatrix},$$

where I_n is the identity matrix of size n and $d(x_i) = x_{i+1} - Ax_i - Bu_i(x_i)$.

Proof *Since $d(\cdot)$ is L_d-Lipschitz, we have by definition for $(x_t, d(x_t)) \in G(d)$ at any timestep t, and $(x_i, d(x_i))$ measured at any $i < t$*

$$\begin{aligned} &\|(d(x_t) - d(x_i))\|^2 \leq L_d^2 \|(x_t - x_i)\|^2, \\ \iff &\begin{bmatrix} x_t \\ d(x_t) \\ 1 \end{bmatrix}^\top Q_L^d(x_i) \begin{bmatrix} x_t \\ d(x_t) \\ 1 \end{bmatrix} \leq 0, \quad \forall (x_t, d(x_t)) \in G(d). \end{aligned}$$

Definition 5.3 (Envelope) *An envelope of a function $f : \mathbb{R}^n \to \mathbb{R}^n$ is defined as any set $\mathbf{E}^f \subseteq \mathbb{R}^n \times \mathbb{R}^n$ with the property*

$$G(f) \subseteq \mathbf{E}^f.$$

Corollary 5.1 *The set defined by*

$$\mathcal{E}(x_i) = \{(x, d) \in \mathbb{R}^{2n} : \begin{bmatrix} x \\ d \\ 1 \end{bmatrix}^\top Q_L^d(x_i) \begin{bmatrix} x \\ d \\ 1 \end{bmatrix} \leq 0\}$$

is an envelope containing the graph of L_d-Lipschitz function $d(\cdot)$ for all timesteps $t \geq 0$, after collecting measurements $(x_i, d(x_i))$ for any $i = 0, 1, \ldots, t-1$.

Lemma 5.1 *Given a sequence of measurements $\{x_i\}_{i=0}^{t-1}$ obtained under dynamics (5.1), we have*

$$G(d) \subseteq \bigcap_{i=0}^{t-1} \mathcal{E}(x_i). \tag{5.5}$$

Proof *See Appendix.*

5.3.2 Uncertainty Estimation at a Given State

We wish to obtain a set where the possible realizations of $d(x_t)$ can lie, which we denote by $\mathcal{D}(x_t)$, for any $x_t \in \mathcal{X}$. Using the collected tuple $(x_i, d(x_i))$ from any time instant $i < t$, we can obtain a set based estimate of the range of possible values of $d(x_t)$, called the *sampled range set* as,

$$\mathcal{S}(x_i, x_t) := \mathcal{E}(x_i)\Big|_{x=x_t} = \{d : \begin{bmatrix} x_t \\ d \\ 1 \end{bmatrix}^\top Q_L^d(x_i) \begin{bmatrix} x_t \\ d \\ 1 \end{bmatrix} \leq 0\},$$

for any $i < t$. As we successively collect $(x_i, d(x_i))$ for $i = \{0, 1, \ldots, t-1\}$, the set of possible values of $d(x_t)$ is obtained and refined with intersection operations as

$$\mathcal{D}(x_t) = \bigcap_{i=0}^{t-1} \mathcal{S}(x_i, x_t) = \bigcap_{i=0}^{t-1} \mathcal{E}(x_i)\Big|_{x=x_t}, \tag{5.6}$$

with the guarantee $d(x_t) \in \mathcal{D}(x_t)$ at any given timestep $t \geq 0$. We further note that the set $\mathcal{D}(x_t)$ is convex, as it is an intersection of convex sets [24].

Proposition 5.2 *Consider a specific state $\tilde{x}$, at time instants t_1 and t_2, with $t_1 < t_2$. Denote them by $\tilde{x}_{t_1}$ and $\tilde{x}_{t_2}$ respectively. Then we have $\mathcal{D}(\tilde{x}_{t_2}) \subseteq \mathcal{D}(\tilde{x}_{t_1})$.*

Proof *Let x_i be the measurements collected at any time instant $i < t$. From (5.6) we see that for any given timestep t, the uncertainty domain $\mathcal{D}(\tilde{x}_t)$ is obtained from successive intersection operations of sampled range sets at $\tilde{x}_t$, for all timesteps until t. Hence, $\mathcal{D}(\tilde{x}_{t_2}) = (\bigcap_{i=0}^{t_1} \mathcal{S}(x_i, \tilde{x}_{t_1})) \bigcap_{i=t_1}^{t_2} \mathcal{S}(x_i, \tilde{x}_{t_2}) = \mathcal{D}(\tilde{x}_{t_1}) \bigcap_{i=t_1}^{t_2} \mathcal{S}(x_i, \tilde{x}_{t_2})$, implying $\mathcal{D}(\tilde{x}_{t_2}) \subseteq \mathcal{D}(\tilde{x}_{t_1})$.*

5.4 The Robust Adaptive MPC Formulation

The main challenges addressed in this section are:

1. Generalizing (5.6) to obtain set based uncertainty bounds along the prediction horizon of the MPC problem (5.3),
2. Posing a tractable optimization problem to solve (5.3) with feasibility guarantees.

5.4.1 Uncertainty Sets Along the MPC Horizon

Recall the definition of Robust Successor States from Definition 2.9. We slightly modify this definition to define the Robust Successor States from any set $\mathcal{A}$ under *any* choice of policy $\pi(\cdot)$ as follows:

$$\begin{aligned}\text{Succ}(\mathcal{A},\mathbb{W}) :=\{x^+ \in \mathcal{X} : \exists x \in \mathcal{A}, \exists u \in \mathcal{U}, \exists w \in \mathbb{W},\\ \text{s.t., } x^+ = Ax + Bu + w\},\end{aligned}$$

with state constraints $\mathcal{X}$ defined in (5.2a). Given any state x_t, an s-procedure based approach to obtain an ellipsoidal outer approximation to $\mathcal{D}(x_t)$, denoted by $E^d(x_t)$, is presented in [24, Section V-A]. We then successively obtain ellipsoidal outer approximations for uncertainty sets $\mathcal{D}(\mathcal{X}_{k|t})$, that is, $E^d(\mathcal{X}_{k|t}) \supseteq \mathcal{D}(\mathcal{X}_{k|t})$, with

$$\mathcal{D}(\mathcal{X}_{k|t}) = \bigcup_{x_{k|t}\in\mathcal{X}_{k|t}} \mathcal{D}(x_{k|t}),$$

where

$$\mathcal{X}_{k|t} \supseteq \text{Succ}(\mathcal{X}_{k-1|t}, E^d(\mathcal{X}_{k-1|t})),\ \forall k = t+1, t+2, \dots, t+N, \tag{5.7a}$$
$$\mathcal{X}_{t|t} = x_t,\ \mathcal{X}_{t+N|t} = \mathcal{X}. \tag{5.7b}$$

Let sets $E^d(\mathcal{X}_{k|t})$ for any $k = \{t, t+1, \dots, t+N\}$ be

$$E^d(\mathcal{X}_{k|t}) := \{d : (d - p^d_{k|t})^\top q^d_{k|t}(d - p^d_{k|t}) \le 1\} = \begin{bmatrix} d \\ 1 \end{bmatrix}^\top \bar{P}^d_{k|t} \begin{bmatrix} d \\ 1 \end{bmatrix} \le 0, \tag{5.8}$$

with $\bar{P}^d_{k|t} = \begin{bmatrix} q^d_{k|t} & -q^d_{k|t}p^d_{k|t} \\ -(p^d_{k|t})^\top q^d_{k|t} & (p^d_{k|t})^\top q^d_{k|t}(p^d_{k|t}) - 1 \end{bmatrix}$, and center $p^d_{k|t} \in \mathbb{R}^n$ and positive definite shape matrix $q^d_{k|t} \in \mathbb{S}^n_{++}$ are decision variables. We consider parametrizations of sets $\mathcal{X}_{k|t}$ as

$$\mathcal{X}_{k|t} := \{x \in \mathbb{R}^n : (x - p^x_{k|t})^\top q^x_{k|t}(x - p^x_{k|t}) \le 1\} = \begin{bmatrix} x \\ 1 \end{bmatrix}^\top \bar{P}^x_{k|t} \begin{bmatrix} x \\ 1 \end{bmatrix} \le 0, \tag{5.9}$$

where $\bar{P}^x_{k|t} = \begin{bmatrix} q^x_{k|t} & -q^x_{k|t}p^x_{k|t} \\ -(p^x_{k|t})^\top q^x_{k|t} & (p^x_{k|t})^\top q^x_{k|t}(p^x_{k|t}) - 1 \end{bmatrix}$ for any $k = \{t, t+1, \dots, t+N\}$. Center $p^x_{k|t} \in \mathbb{R}^n$ and shape matrix $q^x_{k|t} \in \mathbb{S}^n_{++}$ can be successively chosen satisfying (5.7a), with $p^x_{t|t} = x_t$ and $q^x_{t|t} = \text{diag}(\infty, \dots, \infty) \in \mathbb{S}^n_{++}$, if sets $E^d(\mathcal{X}_{k|t})$ are found.

Proposition 5.3 *Using s-procedure, $E^d(\mathcal{X}_{k|t})$ is obtained if the following holds true for some scalars $\{\rho_t^k, \tau_0^k, \tau_1^k, \ldots, \tau_{t-1}^k\} \geq 0$ at each $k = \{t, t+1, \ldots, t+N\}$, for all timesteps $t \geq 0$:*

$$\begin{bmatrix} -\rho_t^k q_{k|t}^x & 0 & \rho_t^k q_{k|t}^x p_{k|t}^x \\ 0 & q_{k|t}^d & -q_{k|t}^d p_{k|t}^d \\ \rho_t^k (p_{k|t}^x)^\top q_{k|t}^x & -(p_{k|t}^d)^\top q_{k|t}^d & (p_{k|t}^d)^\top q_{k|t}^d (p_{k|t}^d) - 1 \\ & & +\rho_t^k - \rho_t^k (p_{k|t}^x)^\top q_t^x (p_{k|t}^x) \end{bmatrix} - \sum_{i=0}^{t-1} \tau_i^k Q_L^d(x_i) \preceq 0. \tag{5.10}$$

Proof *Consider any vector $[x^\top d^\top 1]^\top \in \mathbb{R}^{2n+1}$ such that $x \in E^d(\mathcal{X}_{k|t})$ and $[x^\top d^\top]^\top \in G(d)$. Given that (5.10) is feasible for each prediction instant $k = \{t+1, \ldots, t+N\}$ at any timestep t, we multiply $[x^\top d^\top 1]^\top$ on both sides of (5.10)*

$$-\rho_t^k \begin{bmatrix} x \\ 1 \end{bmatrix}^\top \bar{P}_{k|t}^x \begin{bmatrix} x \\ 1 \end{bmatrix} + \begin{bmatrix} d \\ 1 \end{bmatrix}^\top \bar{P}_{k|t}^d \begin{bmatrix} d \\ 1 \end{bmatrix} - \begin{bmatrix} x \\ d \\ 1 \end{bmatrix}^\top \sum_{i=0}^{t-1} \tau_i^k Q_L^d(x_i) \begin{bmatrix} x \\ d \\ 1 \end{bmatrix} \leq 0,$$

for some $\{\rho_t^k, \tau_0^k, \ldots, \tau_{t-1}^k\} \geq 0$. Now using Corollary 5.1, (5.9) and (5.12), we can infer $\begin{bmatrix} d \\ 1 \end{bmatrix}^\top \bar{P}_{k|t}^d \begin{bmatrix} d \\ 1 \end{bmatrix} \leq 0$.

We reformulate the feasibility problem (5.10) as a Semi-definite Program (SDP) in the Appendix. After finding $E^d(\mathcal{X}_{k|t})$ using (5.10), to efficiently compute (5.7a), we use polytopic outer approximations $\mathcal{P}^d(\mathcal{X}_{k|t}) \supseteq E^d(\mathcal{X}_{k|t})$ instead of $E^d(\mathcal{X}_{k|t})$, given by

$$\begin{aligned} &\mathcal{P}^d(\mathcal{X}_{k|t}) := \{d : H_{k|t}^d d \leq h_{k|t}^d\}, \\ &\forall k = \{t, t+1, \ldots, t+N\}. \end{aligned} \tag{5.11}$$

The choice of this polytope is designer specific.

Remark 5.1 *Consider the state $x_{k|t}$ for prediction step k at timestep t in (5.3). From Proposition 5.3 we know that $d(x_{k|t}) \in \mathcal{D}(\mathcal{X}_{k|t}) \Rightarrow d(x_{k|t}) \in \mathcal{P}^d(\mathcal{X}_{k|t})$, but $d(x_{k|t}) \in \mathcal{P}^d(\mathcal{X}_{k|t}) \nRightarrow d(x_{k|t}) \in \mathcal{D}(\mathcal{X}_{k|t})$. As a consequence, $\mathcal{P}^d(\mathcal{X}_{k|t}) \nsubseteq \mathcal{P}^d(\mathcal{X}_{k|t-1})$ is possible. Hence, for ensuring recursive feasibility of solved MPC problem (detailed in Theorem 5.1), we impose constraints in (5.3) robustly for all $d(x_{k|t})$ satisfying*

$$\begin{aligned} &d(x_{k|t}) \in \mathcal{P}^d(\mathcal{X}_{k|t}) \cap \mathcal{P}^d(\mathcal{X}_{k|t-1}), \\ &\forall k \in \{t, \ldots, t+N-1\}, \end{aligned} \tag{5.12}$$

with the initialization

$$\{q_{-1|-1}^d, q_{0|-1}^d, \ldots, q_{N-2|-1}^d\} = \{0_{n\times n}, \ldots, 0_{n\times n}\} \in \mathbb{R}^{n \times Nn}.$$

5.4.2 Terminal Conditions

Terminal set $\mathcal{X}_N$ is chosen as the maximal robust positive invariant set [93, 94] obtained with a state feedback controller $u = Kx$, dynamics (5.1) and constraints (5.2), with properties

$$\begin{aligned} &\mathcal{X}_N \subseteq \{x | H_x x \leq h_x,\ H_u K x \leq h_u\}, \\ &(A+BK)x + d(x) \in \mathcal{X}_N, \\ &\forall x \in \mathcal{X}_N,\ \forall d(x) \in \mathcal{P}^d(\mathcal{X}). \end{aligned} \tag{5.13}$$

Fixed point iteration algorithms to numerically compute (5.13) can be found in [6, 75].

5.4.3 Tractable MPC Problem

The tractable MPC optimization problem at timestep t is given by:

$$\begin{aligned} \min_{\mathbf{M}_t, \mathbf{v}_t} \quad & \sum_{k=t}^{t+N-1} (\bar{x}_{k|t}^\top Q \bar{x}_{k|t} + v_{k|t}^\top R v_{k|t}) + \bar{x}_{t+N|t}^\top P_N \bar{x}_{t+N|t} \\ \text{s.t.,} \quad & x_{k+1|t} = Ax_{k|t} + Bu_{k|t}(x_{k|t}) + d(x_{k|t}), \\ & \bar{x}_{k+1|t} = A\bar{x}_{k|t} + Bv_{k|t} + \bar{d}_{k|t}, \\ & u_{k|t}(x_{k|t}) = \sum_{l=t}^{k-1} M_{k,l|t} d(x_{l|t}) + v_{k|t}, \\ & H_x x_{k+1|t} \leq h_x,\ H_u u_{k|t}(x_{k|t}) \leq h_u, \\ & \forall d(x_{k|t}) \in \mathcal{P}^d(\mathcal{X}_{k|t}) \cap \mathcal{P}^d(\mathcal{X}_{k|t-1}), \\ & \forall k = \{t, \ldots, t+N-1\}, \\ & x_{t+N|t} \in \mathcal{X}_N,\ \forall d(x_{N|t}) \in \mathcal{P}^d(\mathcal{X}), \\ & x_{t|t} = x_t,\ \bar{x}_{t|t} = x_t,\ \bar{d}_{k|t} \in \mathcal{P}^d(\mathcal{X}_{k|t}), \end{aligned} \tag{5.14}$$

where the parameters $\{p_{k|t}^d, q_{k|t}^d\}$ for $k = \{t, t+1, \ldots, t+N\}$, that is, uncertainty containment ellipses in (5.14), are computed before solving (5.14) at each timestep t, by finding solutions of (5.10). Nominal uncertainty estimate $\bar{d}_{k|t}$ is chosen as the Chebyshev center (i.e., center of the largest volume ℓ_2 ball in a set) of $\mathcal{P}^d(\mathcal{X}_{k|t})$. After solving (5.14) at timestep t, in closed-loop we apply

$$u_t(x_t) = v_{t|t}^\star, \tag{5.15}$$

to system (5.1) and then resolve (5.14) at $t+1$.

Remark 5.2 *Terminal set $\mathcal{X}_N$ might be empty initially, due to conservatism resulting from a large volume of the set $\mathcal{P}^d(\mathcal{X})$. As more data is collected and the graph of $d(\cdot)$ is refined as in (5.5)–(5.6), $E^d(\mathcal{X})$, and so $\mathcal{P}^d(\mathcal{X})$ is refined with new data by solving (5.10) (for only*

$k = t + N$, if data collected until instant t) with an updated $Q_L^d(\cdot)$. This eventually results in a nonempty $\mathcal{X}_N$. Once (5.14) is feasible with this $\mathcal{X}_N$, during the control process one may further update and enlarge $\mathcal{X}_N$ to lower conservatism of (5.14).

Algorithm 3 Robust Adaptive MPC with Additive Lipschitz Uncertainty

Initialize: $\mathcal{P}^d(\mathcal{X}) = \mathbb{R}^n$; $j = 0$;

begin exploration (offline)

while $\mathcal{X}_N$ is empty **do**
  Apply exploration inputs u_j to (5.1). Collect $(x_j, d(x_j))$ at $j+1$. Set $j = j + 1$;
  Solve (5.10) with $k = j + N$ to get $\mathcal{P}^d(\mathcal{X})$. Compute $\mathcal{X}_N$ from (5.13);
end while

end exploration **set** $j_{\max} \equiv t = 0$.

begin control process (online)

while during control for $t \geq 0$ **do**
  Obtain $\mathcal{P}^d(\mathcal{X}_{k|t})$ for $k = \{t, t+1, \ldots, t+N-1\}$ from feasibility of (5.10);
  if larger $\mathcal{X}_N$ desired **then** Update $\mathcal{P}^d(\mathcal{X})$ from (5.10) (with $k = t + N$). Update $\mathcal{X}_N$;
  end if
  Solve (5.14) and apply MPC (5.15) to (5.1);
end while

Theorem 5.1 *Let optimization problem (5.14) be feasible at timestep $t = 0$. Assume the state dependent uncertainty $d(\cdot)$ bounds along the horizon are obtained using (5.10), (5.7), and (5.11). Then, (5.14) remains feasible at all timesteps $t \geq 0$, if the state x_t is obtained by applying the closed-loop MPC control law (5.15) to system (5.1).*

Proof *Let the optimization problem (5.14) be feasible at timestep t. Let us denote the corresponding optimal input policies as $[u^\star_{t|t}(\cdot), u^\star_{t+1|t}(\cdot), \cdots, u^\star_{t+N-1|t}(\cdot)]$. Assume the MPC controller $u^\star_{t|t}$ is applied to (5.1) in closed-loop and $E^d(\mathcal{X}_{k|t+1})$ for $k = \{t+1, t+2, \ldots, t+N+1\}$ are obtained according to (5.10), (5.11) and (5.7). Consider a candidate policy sequence at the next time instant as:*

$$\Pi_{t+1}(\cdot) = [u^\star_{t+1|t}(\cdot), \ldots, u^\star_{t+N-1|t}(\cdot), Kx_{t+N|t+1}]. \tag{5.16}$$

From (5.12) and Proposition 5.2 we conclude the sequence $[u^\star_{t+1|t}(\cdot), u^\star_{t+2|t}(\cdot), \ldots, u^\star_{t+N-1|t}(\cdot)]$ is an $(N-1)$ step feasible policy sequence at $t+1$ (excluding terminal condition), since at previous timestep t, it robustly satisfied all stage constraints in (5.14). With this feasible policy sequence, $x_{t+N|t+1} \in \mathcal{X}_N$. From (5.13) we conclude that (5.16) ensures the satisfaction of $x_{t+N+1|t+1} \in \mathcal{X}_N$.

5.5 Numerical Example

In this section we demonstrate both the aspects of exploration and robust control of our robust adaptive MPC, highlighted in Algorithm 3. We wish to compute feasible solutions to the following infinite horizon control problem

$$\begin{aligned} \min_{u_0,u_1(\cdot),\dots} \quad & \sum_{t\geq 0} \bar{x}_t^\top Q \bar{x}_t + u_t^\top(\bar{x}_t) R u_t(\bar{x}_t) \\ \text{s.t.,} \quad & x_{t+1} = Ax_t + Bu_t(x_t) + 0.05 \begin{bmatrix} \tan^{-1}(x_t(1)) \\ x_t(2) \end{bmatrix} \\ & \begin{bmatrix} -1 \\ -1 \\ -4 \end{bmatrix} \leq \begin{bmatrix} x_t \\ u_t(x_t) \end{bmatrix} \leq \begin{bmatrix} 1.5 \\ 3 \\ 1 \end{bmatrix}, \\ & \forall d(x_t) \in \mathcal{D}(x_t), \\ & x_0 = \bar{x}_0 = x_S, \ t = 0, 1, \dots, \end{aligned} \tag{5.17}$$

with initial state $x_S = [-1, 2]^\top$, where

$$A = \begin{bmatrix} 1.2 & 1.5 \\ 0 & 1.3 \end{bmatrix}, \ B = [0, 1]^\top.$$

Algorithm 3 is implemented with a control horizon of $N = 3$, and the feedback gain K in (5.13) is chosen to be the optimal LQR gain for system $x^+ = (A + BK)x$ with $Q = 10I_2$ and $R = 2$. The source code is available in the repository https://github.com/monimoyb/AMPC_StateDepUncertainty.

5.5.1 Exploration for Uncertainty Learning

We initialize $\mathcal{P}^d(\mathcal{X}) = \mathbb{R}^n$, resulting in an empty terminal set $\mathcal{X}_N$ in (5.14). In this section, we present the ability of Algorithm 3 to explore the state-space with randomly generated inputs $u_j \sim \mathcal{N}(0, 1)$, in order to eventually obtain a nonempty $\mathcal{X}_N$ for starting the control process. Let the time indices during exploration phase be denoted by j. Fig. 5.1 shows the sets $E^d(x)$ at four fixed *query* points

$$x_j = \{[-1, 2]^\top, [1, 1]^\top, [-1, 1]^\top, [-2, -1]^\top\},$$

as data is collected until time instant j. This can be obtained from the feasibility of (5.10) (with $k = j$). As j increases, $E^d(x)$ for each x is contained in the successive intersections of ellipsoids, from (5.6). The intersection shrinks for all points, as claimed in Proposition 5.2. This is seen in Fig. 5.1, which indicates improved information of $\mathcal{D}(x)$ with added data, for all $x \in \mathcal{X}$. At $j_{\max} = 30$, a nonempty $\mathcal{X}_N$ is obtained in Fig. 5.2. This is when we start the control process and set timestep $t = 0$.

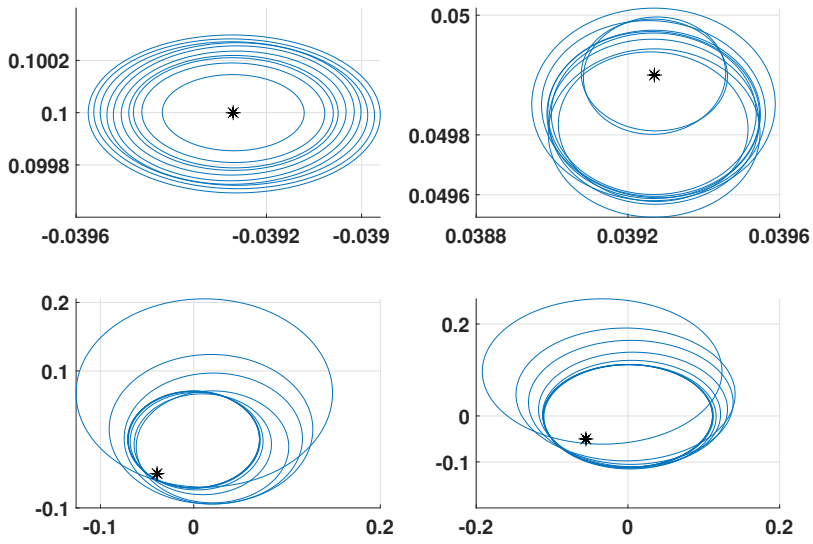


Figure 5.1: Uncertainty bound $\mathcal{D}(x)$ estimation at *query* points with successive intersection of ellipses obtained from measured data. Star ($\star$) denotes the true value of $d(x)$, lying in the intersection.

5.5.2 Robust Constraint Satisfaction

If the MPC problem (5.14) is feasible for parameters defined in (5.17), it ensures robust satisfaction of constraints in (5.17) for all timesteps $t > 0$. This is highlighted with a realized trajectory in Fig. 5.3. Furthermore, the terminal set is recomputed and improved at a $t > 0$ with (5.13), having refined $\mathcal{P}^d(\mathcal{X})$ estimation (rectangles with sides of length equal to major and minor axes of $E^d(\cdot)$) from (5.10) (with $k = t + N$). The set grows, as seen in Fig. 5.2, resulting in lesser conservatism of (5.14).

5.6 Chapter Summary

We proposed a robust adaptive MPC framework to achieve robust satisfaction of state and input constraints for uncertain linear time-invariant systems. The system uncertainty is additive, and assumed state dependent and globally Lipschitz. An envelope containing the uncertainty range is constructed with Quadratic Constraints (QCs), and is refined with data as the system explores the state-space. Upon collection of sufficient data, the system is able to solve a robust MPC problem for all times from a given initial state. The algorithm further reduces its conservatism by incorporating online model adaptation during control.

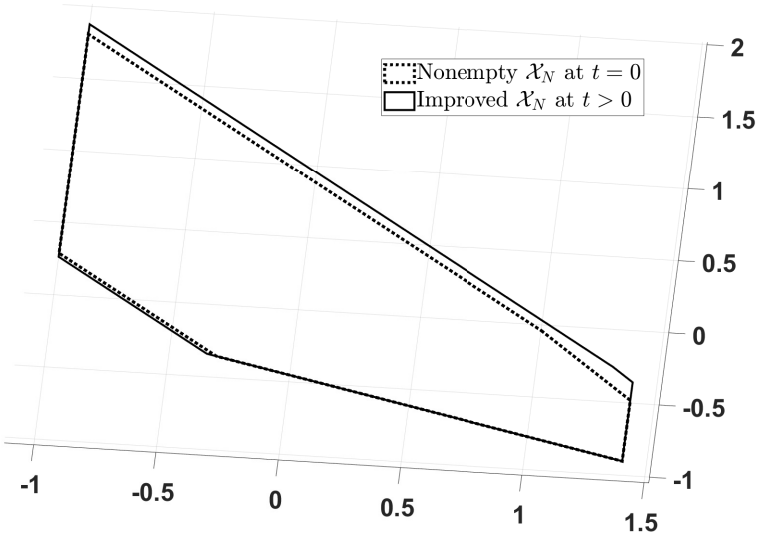


Figure 5.2: Terminal set construction. The set grows as estimation of $d(x)$ is improved from measurements.

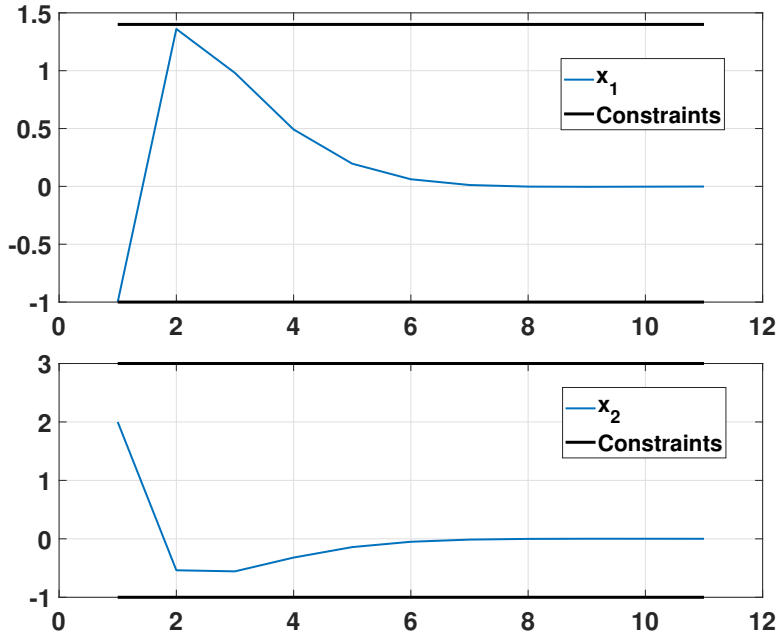


Figure 5.3: State trajectory with robust constraint satisfaction.

Appendix

Proof of Lemma 5.1

For any $(x, d(x)) \in G(d)$, we have from the Lipschitz inequality,

$$\|d(x) - d(y)\| \leq L_d \|x - y\|, \quad \forall y \in \mathcal{X},$$

and choosing $y = x_i$ for $i = 0, 1, \ldots t-1$ in the above inequality, using Corollary 5.1 yields,

$$\begin{aligned} &(x, d(x)) \in \mathcal{E}(x_i), \quad \forall i = 0, 1, \ldots t-1, \\ \Rightarrow &(x, d(x)) \in \bigcap_{i=0}^{t-1} \mathcal{E}(x_i). \end{aligned}$$

Since the above is true for any $(x, d(x)) \in G(d)$, we can conclude that

$$G(d) \subseteq \bigcap_{i=0}^{t-1} \mathcal{E}(x_i).$$

SDP for Solving (5.10)

For all $k = \{t+1, \ldots, t+N\}$, along MPC horizon, let us use the variable nomenclature $p(\mathcal{X}_{k|t}) = -\rho_t^k q_{k|t}^x + \sum_{i=0}^{t-1} \tau_i^k L_d^2 I_n$, $q(\mathcal{X}_{k|t}) = \rho_t^k (q_{k|t}^x)^\top p_{k|t}^x - \sum_{i=0}^{t-1} \tau_i^k L_d^2 x_i$, $r(\mathcal{X}_{k|t}) = -\sum_{i=0}^{t-1} \tau_i^k I_n$, $s(\mathcal{X}_{k|t}) = \sum_{i=0}^{t-1} \tau_i^k d(x_i)$, and $t(\mathcal{X}_{k|t}) = \rho_t^k \Big(1 - (p_{k|t}^x)^\top q_{k|t}^x p_{k|t}^x\Big) - \sum_{i=0}^{t-1} \tau_i^k \Big(-L_d^2 x_i^\top x_i + d^\top(x_i) d(x_i)\Big) - 1$. Finding the minimum trace ellipsoid satisfying (5.10) is posed as an SDP [45, Section 11.4] as:

$$\begin{aligned} \min_{\xi} \quad & \text{trace}((q_{k|t}^d)^{-1}) \\ \text{s.t.,} \quad & \begin{bmatrix} p(\mathcal{X}_{k|t}) & \mathbf{0} & q(\mathcal{X}_{k|t}) & \mathbf{0} \\ \mathbf{0} & r(\mathcal{X}_{k|t}) & s(\mathcal{X}_{k|t}) & -I_n \\ q^\top(\mathcal{X}_{k|t}) & s^\top(\mathcal{X}_{k|t}) & t(\mathcal{X}_{k|t}) & (p_{k|t}^d)^\top \\ \mathbf{0} & -I_n & p_{k|t}^d & -(q_{k|t}^d)^{-1} \end{bmatrix} \preceq 0, \\ & \rho_t^k \geq 0, \tau_i^k \geq 0, \ q_{k|t}^d \succ 0, \ \forall i = \{0, 1, \ldots, t-1\}, \end{aligned}$$

with $\xi = \{q_{k|t}^d, p_{k|t}^d, \rho_t^k, \tau_0^k, \ldots, \tau_{t-1}^k\}$ and $\mathbf{0} \in \mathbb{R}^{n \times n}$.

Chapter 6

Learning Parametric Model Uncertainty in Robust MPC

This chapter is based on the published work [31]. In this chapter, we propose a unified and tractable *adaptive MPC* framework for linear systems of the form (M2) in Chapter 2. We specifically focus on systems (2.15)-(2.16). However, as a first simplified step, we consider a linear time-invariant system with known system matrices (A, B) that is subject to bounded additive uncertainty, which is composed of: (i) a disturbance, and (ii) an unknown, but bounded time-varying parametric offset. The presence of such an unknown offset is common in various practical systems, e.g., the steering system of a vehicle [95]. We learn and refine the feasible domain of this parameter using collected data. The results are then extended for systems of the form (2.15)-(2.16) in Section 6.7.

6.1 Summary of Contributions

Given an initial estimate of the additive offset's domain, we iteratively refine it using a Set Membership Method based approach [22], as new data becomes available. In order to design an MPC controller with the unknown offset, we make sure the constraints on states and inputs are satisfied for all feasible offsets at a time instant. Here a "feasible offset" is an offset belonging to the current estimation of the offset's domain. As the feasible offset domain is updated with data progressively, we obtain an on-line adaptation in the MPC algorithm. Furthermore, the offset uncertainty present in the system is considered time-varying and its maximum rate of change is assumed bounded and known [96, 97]. The main contributions of this chapter can be summarized as follows:

- We propose a Set Membership Method based model adaptation algorithm to estimate and update the time-varying offset uncertainty, using a so-called Feasible Parameter Set. The model adaptation algorithm guarantees containment of the true offset uncertainty in the Feasible Parameter Set at all times. This extends the works of [21, 22, 97, 12] to time-varying model uncertainties in state-space.

- We propose an adaptive MPC algorithm for systems perturbed by such an additive time-varying offset uncertainty and a disturbance. The framework handles robust constraints on system states with hard input constraints, while using data to progressively obtain offset uncertainty adaptation. With appropriately chosen terminal conditions, we guarantee recursive feasibility and Input to State Stability (ISS) of the proposed adaptive MPC algorithm, which is an addition to the work of [7, 98, 8, 9]. Compared to [99, 100, 96], the computation of terminal invariant set is simpler, as we focus on linear systems. Moreover, as opposed to [20, 101, 8], we utilize the model adaptation information in real-time for modifying the imposed constraints in the MPC problem.
- We extend our results to linear parameter varying systems of the form (2.15)-(2.16). Merging the parameter adaptation algorithm with the robust MPC from Chapter 3 we demonstrate computational efficiency of the adaptive MPC and successive enlargement of the ROA in a numerical example.

6.2 Problem Formulation

Before considering parametric uncertainty in models of the form (M2) shown in Chapter 2, we consider a simplified case of an uncertain linear time-invariant system of the form:

$$x_{t+1} = Ax_t + Bu_t + E\theta_t^a + w_t, \tag{6.1}$$

where $x_t \in \mathbb{R}^n$ is the state at timestep t, $u_t \in \mathbb{R}^m$ is the input, and A and B are known system matrices of appropriate dimensions. At each timestep t, the system is affected by disturbance $w_t \in \mathbb{W} \subset \mathbb{R}^n$. For simplicity, $\mathbb{W}$ is assumed to be a hyperrectangle containing zero as:

$$\mathbb{W} = \{w : -\bar{w} \leq w \leq \bar{w}\}. \tag{6.2}$$

We also consider the presence of $\theta_t^a \in \mathbb{R}^p$, a bounded, time-varying offset uncertainty, which enters the system through the constant known matrix $E \in \mathbb{R}^{n\times p}$.

Assumption 6.1 *We assume the true offset θ_t^a to be time-varying. The bounds on the rate of change of this offset are known and given by $\theta_t^a - \theta_{t-1}^a = \Delta\theta_t^a \in \mathcal{P}$, for all $t \geq 0$, where the set*

$$\mathcal{P} = \{\Delta\theta^a \in \mathbb{R}^p : K^\theta \Delta\theta^a \leq l^\theta, K^\theta \in \mathbb{R}^{r_\theta \times p}, l^\theta \in \mathbb{R}^{r_\theta}\}. \tag{6.3}$$

Assumption 6.2 *We also assume that the true offset θ_t^a lies within a known and nonempty polytope Ω at all times, which contains zero in its interior. That is,*

$$\theta_t^a \in \Omega, \ \forall t \geq 0, \ \textit{where, } \Omega = \{\theta : \mathcal{H}_0^\theta \theta \leq h_0^\theta\}, \tag{6.4}$$

for matrices $\mathcal{H}_0^\theta \in \mathbb{R}^{r_0 \times p}$ and $h_0^\theta \in \mathbb{R}^{r_0}$.

We assume that the offset θ_t^a is *not known exactly*. Therefore, we propose a parameter estimation framework to refine our knowledge of θ_t^a as more data is collected, thus introducing *adaptation*.

6.3 Parametric Uncertainty Adaptation

The domain of feasible offset θ_t^a is denoted by Θ_t at timestep t, and is called the *Feasible Parameter Set* [22]. The goal is to ensure that constraints (6.8a) are satisfied for all $\theta_t \in \Theta_t$. This guarantees constraint satisfaction in presence of the true unknown offset $\theta_t^a \in \Theta_t$. Our initial estimate for Θ_0 is Ω from Assumption 6.2, i.e., $\Theta_0 = \Omega$. The Feasible Parameter Set is then adapted at every timestep as new measurements are available, utilizing Assumption 6.1 and Assumption 6.2. Based on only the measurements at timestep t, we denote the potential domain of the feasible offset at timestep t, $\mathcal{S}_t^t$ as:

$$\mathcal{S}_t^t = \{\theta_t \in \mathbb{R}^p : -\bar{w} + \underline{\nu} \leq -x_t + Ax_{t-1} + Bu_{t-1} + E\theta_t \leq \bar{w} + \bar{\nu}\},$$

where bounds $\bar{w}$ are given by (6.2), and from (6.3), we apply:

$$\begin{aligned} \underline{\nu} &= \min_{\nu}\{E\nu : K^\theta \nu \leq l^\theta\}, \\ \bar{\nu} &= \max_{\nu}\{E\nu : K^\theta \nu \leq l^\theta\}. \end{aligned} \tag{6.5}$$

Now, for any $q \leq t$, the feasible set of offsets for timestep t, based on information until timestep q, is obtained as:

$$\mathcal{S}_t^q = \{\theta_t \in \mathbb{R}^p : -\bar{w} + (t-q+1)\underline{\nu} \leq -x_q + Ax_{q-1} + Bu_{q-1} + E\theta_t \leq \bar{w} + (t-q+1)\bar{\nu}\},$$

Using all the above information until timestep t, we obtain the Feasible Parameter Set at timestep t, as:

$$\Theta_t = \Omega \cap \Big(\bigcap_{q=1,2,\ldots,t} \mathcal{S}_t^q \Big).$$

The above Feasible Parameter Set at timestep t can be written as:

$$\Theta_t = \{\theta_t \in \mathbb{R}^p : \mathcal{H}_t^\theta \theta_t \leq h_t^\theta\}, \tag{6.6}$$

where $\mathcal{H}_t^\theta \in \mathbb{R}^{r_t \times p}$ and $h_t^\theta \in \mathbb{R}^{r_t}$, $r_t = r_0 + 2t$ is the number of facets in the Feasible Parameter Set polytope Θ_t at any given t. As new data is obtained at the next timestep $(t+1)$, it can be proven that [97]:

$$\begin{aligned} \mathcal{H}_{t+1}^\theta &= [(\mathcal{H}_t^\theta)^\top, -E^\top, +E^\top]^\top \in \mathbb{R}^{r_{t+1} \times p}, \\ h_{t+1}^\theta &= \begin{bmatrix} h_t^\theta + \Delta h_t^\theta \\ -x_{t+1} + Ax_t + Bu_t + \bar{w} - \underline{\nu} \\ x_{t+1} - Ax_t - Bu_t + \bar{w} + \bar{\nu} \end{bmatrix} \in \mathbb{R}^{r_{t+1}}, \\ \Delta h_t^\theta &= \left[\mathbf{0}_{r_0}^\top, -\underline{\nu}^\top, \bar{\nu}^\top, \ldots, -\underline{\nu}^\top, \bar{\nu}^\top\right]^\top \in \mathbb{R}^{r_t}. \end{aligned} \tag{6.7}$$

Proposition 6.1 *Assume that (6.2) and Assumption 6.1 hold. Then the Feasible Parameter Set obtained using (6.6)–(6.7) is nonempty and contains the true offset at all times, i.e., $\Theta_t \neq \emptyset$ and $\theta_t^a \in \Theta_t$ for all $t \geq 0$.*

Proof *See Appendix.*

6.4 The Robust Adaptive MPC

In this section we present formulation of the proposed robust adaptive MPC algorithm. We study robust constraints on states and hard constraints on inputs. We define $C \in \mathbb{R}^{s\times n}$, $G \in \mathbb{R}^{s\times n}$, $D \in \mathbb{R}^{s\times m}$, $b \in \mathbb{R}^s$. We can then write the constraints $\forall t \geq 0$ as:

$$\mathbb{Z} := \{(x, u) : Cx + Du \leq b\}. \tag{6.8a}$$

We assume the above state and input constraint sets are compact and they contain the origin. This assumption is key for the stability proof in Section 6.5.

6.4.1 Robust MPC Problem

The MPC controller has to solve the following finite horizon robust optimal control problem at each timestep:

$$\begin{aligned}
\min_{U_t(\cdot)} \ & \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})) + Q(\bar{x}_{t+N|t}) \\
\text{s.t.}, \ & x_{k+1|t} = Ax_{k|t} + Bu_{k|t}(x_{k|t}) + E\theta_{k|t} + w_{k|t}, \\
& \bar{x}_{k+1|t} = A\bar{x}_{k|t} + Bu_{k|t}(\bar{x}_{k|t}) + E\bar{\theta}_t, \\
& Cx_{k|t} + Du_{k|t}(x_{k|t}) \leq b, \\
& x_{t+N|t} \in \mathcal{X}_N, \\
& \forall \theta_{k|t} \in \Theta_{k|t}, \ \forall w_{k|t} \in \mathbb{W}, \\
& \forall k = \{t, \dots, t+N-1\}, \\
& x_{t|t} = x_t, \ \bar{x}_{t|t} = x_t, \bar{\theta}_t \in \Omega,
\end{aligned} \tag{6.9}$$

with $U_t(\cdot) = \{u_{t|t}, u_{t+1|t}(\cdot), \dots, u_{t+N-1|t}(\cdot)\}$, where x_t is the measured state at timestep t, $x_{k|t}$ is the prediction of state at timestep k, obtained by applying predicted input policies $\{u_{t|t}, \dots, u_{k-1|t}(x_{k-1|t})\}$ to system (6.1), and $\{\bar{x}_{k|t}, u_{k|t}(\bar{x}_{k|t})\}$ denote the disturbance-free nominal state and corresponding input respectively. We use a nominal point estimate of offset, $\bar{\theta}_t \in \Omega$ to propagate the nominal trajectory. The predicted Feasible Parameter Sets $\Theta_{k|t}$ are elaborated in the following section. Notice, the above minimizes the nominal cost, comprising of positive definite stage cost $\ell(\cdot, \cdot)$ and terminal cost $Q(\cdot)$ functions. The terminal constraint $\mathcal{X}_N$ and terminal cost $Q(\cdot)$ are introduced to ensure feasibility and stability properties of the MPC controller [5, 6], as we highlight in Section 6.5.

Remark 6.1 *One may design point estimates $\bar{\theta}_t$ of the offset for performance improvement, i.e., lower cost in (6.9). Following [92], one option is to construct the nominal offset estimate $\bar{\theta}_t$ recursively with Least Mean Square filter as*

$$\tilde{\theta}_t = \bar{\theta}_{t-1} + \mu E^\top (x_t - \bar{x}_{t|t-1}), \tag{6.10a}$$

$$\bar{\theta}_t = \mathrm{Proj}_\Omega(\tilde{\theta}_t), \tag{6.10b}$$

where $\mathrm{Proj}(\cdot)$ *denotes the Euclidean projection operator, and scalar* $\mu \in \mathbb{R}$ *can be chosen such that* $\frac{1}{\mu} > \|E\|^2$.

Proposition 6.2 *If* $\sup_{t\geq 0} \|x_t\| < \infty$ *and* $\sup_{t\geq 0} \|u_t\| < \infty$, *then* $\bar{\theta}_t \in \Omega$ *and*

$$\sup_{\tilde{m}\in\mathbb{N}, w_t\in\mathbb{W}, \bar{\theta}_0\in\Omega} \frac{\sum_{t=0}^{\tilde{m}} \|\tilde{x}_{t+1|t}\|^2}{\frac{1}{\mu}\|\bar{\theta}_0 - \theta_0^a\|^2 + \sum_{t=0}^{\tilde{m}} \|w_t\|^2} \leq 1, \tag{6.11}$$

where $\tilde{x}_{t+1|t} = Ax_t + Bu_t - \bar{x}_{t+1|t}$ *is the one step prediction error, ignoring the effect of* w_t *in closed-loop, and* $\mathbb{N}$ *denotes the set of natural numbers.*

Proof *See Appendix.*

With bound (6.11) on prediction error, finite gain ℓ_2 *stability of the resulting MPC algorithm can be trivially proven by following [92, Theorem 14], [75, Theorem 3.2]. However, since we only focus on the robust constraint satisfaction aspect of (6.9), we will use the nominal offset estimate* $\bar{\theta}_t = \mathbf{0}_{p\times 1}$ *for all* $t \geq 0$ *in the subsequent sections.*

6.4.2 Predicted Feasible Parameter Sets

These *Predicted Feasible Parameter Sets* are constructed along an MPC horizon at timestep t, when the measurement at next timestep $(t+1)$ is yet to be available.

Definition 6.1 *(Predicted Feasible Parameter Sets) The* Predicted Feasible Parameter Sets *at any timestep* t, *are the predicted feasible domains of the true offset* θ^a *over a prediction horizon of length* N, *based on the information until timestep* t. *These sets are denoted as* $\Theta_{k|t} = \{\theta \in \mathbb{R}^p : \mathcal{H}^\theta_{k|t}\theta \leq h^\theta_{k|t}\}$ *for all* $k \in \{t, t+1, \ldots, t+N-2\}$, *where*

$$\mathcal{H}^\theta_{k+1|t} = \mathcal{H}^\theta_{k|t} \in \mathbb{R}^{r_t \times p}, \tag{6.12a}$$

$$h^\theta_{k+1|t} = h^\theta_{k|t} + \left[\mathbf{0}^\top_{r_0}, -\underline{\nu}^\top, \bar{\nu}^\top, .., -\underline{\nu}^\top, \bar{\nu}^\top\right]^\top, \tag{6.12b}$$

with the terminal condition,

$$\Theta_{t+N|t} = \Omega, \tag{6.13}$$

where Ω *is defined in Assumption 6.2.*

In principle, the Predicted Feasible Parameter Sets in (6.12) are formed after measuring x_t at any timestep t, and expanding the obtained (from (6.6)) Feasible Parameter Set Θ_t over the entire horizon of length N, incorporating parameter rate bounds (6.3).

Proposition 6.3 *The Predicted Feasible Parameter Sets satisfy the property* $\Theta_{k|t+1} \subseteq \Theta_{k|t}$, *for all* $k \in \{t+1, t+2, \ldots, t+N\}$.

Proof *See Appendix.*

6.4.3 Control Policy

We restrict ourselves to the affine disturbance feedback parametrization, as per (P2) in Chapter 2. For all $k \in \{t, \ldots, t+N-1\}$ over the MPC horizon (of length N), the control policy is given as:

$$u_{k|t}(x_{k|t}) = \sum_{j=t}^{k-1} M_{k,j|t}(w_{j|t} + E\theta_{j|t}) + v_{k|t}, \tag{6.14}$$

where $M_{k|t}$ are the planned feedback gains at timestep t and $v_{k|t}$ are the auxiliary inputs. Let us define $\mathbf{w}_t = [w_{t|t}^\top, \ldots, w_{t+N-1|t}^\top]^\top \in \mathbb{R}^{nN}$, $\boldsymbol{\theta}_t = [\theta_{t|t}^\top, \ldots, \theta_{t+N-1|t}^\top]^\top \in \mathbb{R}^{pN}$ and $\mathbf{E} = \mathrm{diag}(E, E, \ldots, E) \in \mathbb{R}^{nN \times pN}$. Then the sequence of predicted inputs from (6.14) can be stacked together and compactly written as $\mathbf{u}_t = \mathbf{M}_t(\mathbf{w}_t + \mathbf{E}\boldsymbol{\theta}_t) + \mathbf{v}_t$ at any timestep t, where $\mathbf{M}_t \in \mathbb{R}^{mN \times nN}$ and $\mathbf{v}_t \in \mathbb{R}^{mN}$ are shown in (2.28).

6.4.4 Tractable Reformulation

Using Section 6.4.2 and Section 6.4.3, we solve the following tractable reformulation of robust MPC problem (6.9):

$$\begin{aligned}
J_R^\star(t, x_t) := \\
\min_{\mathbf{M}_t, \mathbf{v}_t} \quad & \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}, v_{k|t}) + Q(\bar{x}_{t+N|t}) \\
\text{s.t.,} \quad & x_{k+1|t} = Ax_{k|t} + Bu_{k|t} + E\theta_{k|t} + w_{k|t}, \\
& \bar{x}_{k+1|t} = A\bar{x}_{k|t} + Bv_{k|t}, \\
& u_{k|t} = \sum_{j=t}^{k-1} M_{k,j|t}(w_{j|t} + E\theta_{j|t}) + v_{k|t}, \\
& Cx_{k|t} + Du_{k|t} \leq b, \\
& x_{t+N|t} \in \mathcal{X}_N, \\
& \forall \theta_{k|t} \in \Theta_{k|t}, \ \forall w_{k|t} \in \mathbb{W}, \\
& \forall k = \{t, \ldots, t+N-1\}, \\
& x_{t|t} = x_t, \ \bar{x}_{t|t} = x_t.
\end{aligned} \tag{6.15}$$

We use state feedback to construct terminal set $\mathcal{X}_N = \{x \in \mathbb{R}^n : Y_R x \leq z_R,\ Y_R \in \mathbb{R}^{r_R \times n},\ z_R \in \mathbb{R}^{r_R}\}$, which is the maximal robust positive invariant set obtained with a state feedback controller $u = Kx$, dynamics (6.1) and constraints (6.8a). This set has the properties:

$$\begin{aligned} &\mathcal{X}_N \subseteq \{x | (x, Kx) \in \mathbb{Z}\}, \\ &(A + BK)x + w + E\theta \in \mathcal{X}_N, \\ &\forall x \in \mathcal{X}_N,\ \forall w \in \mathbb{W},\ \forall \theta \in \Omega. \end{aligned} \tag{6.16}$$

Notice that (6.15) is a *time-varying* convex optimization problem with $\infty-$number of constraints. An efficient way to reformulate (6.15) is shown in the Appendix. After solving (6.15), in closed-loop, we apply,

$$u_t(x_t) = u^\star_{t|t} = v^\star_{t|t} \tag{6.17}$$

to system (6.1). We then resolve the problem again at the next $(t+1)$-th timestep, yielding a receding horizon strategy.

Algorithm 4 Robust Adaptive MPC with Time-Varying Parametric Additive Uncertainty

1: Set $t = 0$; initialize Feasible Parameter Set $\Theta_0 = \Omega$;
2: Compute the parameter rate of change bounds $\underline{\nu}$ and $\bar{\nu}$ from (6.5);
3: Form Predicted Feasible Parameter Sets $\Theta_{k|t}$ for $k = \{t, \ldots, t+N\}$ using (6.12) and (6.13);
4: Compute $v^\star_{t|t}$ from (6.15) and apply $u_t = v^\star_{t|t}$ to (6.1);
5: Obtain x_{t+1}, and update Θ_{t+1} as given in (6.7);
6: Set $t = t + 1$, and return to step 3.

6.5 Feasibility and Stability Guarantees

In this section we discuss feasibility and stability properties of Algorithm 4.

Assumption 6.3 *The stage cost $\ell(\cdot,\cdot)$ in (6.15) is chosen as $\ell(\bar{x}_{k|t}, v_{k|t}) = \bar{x}^\top_{k|t} P \bar{x}_{k|t} + v^\top_{k|t} R v_{k|t}$ for some $P = P^\top \succ 0$ and $R = R^\top \succ 0$, which is continuous and positive definite.*

Assumption 6.4 *The terminal cost $Q(\cdot)$ in (6.15) is chosen as a Lyapunov function in the terminal set $\mathcal{X}_N$ for the nominal closed-loop system $x^+ = (A + BK)x$, for all $\bar{x} \in \mathcal{X}_N$. That is, $Q((A+BK)x) - Q(x) \leq -x^\top (P + K^\top R K)x$.*

6.5.1 Feasibility

Theorem 6.1 *Let Assumptions 6.1-6.2 hold and consider the robust optimization problem (6.15). Let this optimization problem be feasible at timestep $t = 0$ with $\Theta_0 = \Omega$ with Ω*

defined in Assumption 6.2. Assume the Feasible Parameter Set Θ_t in (6.15) is adapted based on (6.6)-(6.7). Then, (6.15) remains feasible at all timesteps $t \geq 0$, if the state x_t is obtained by applying the closed-loop MPC control law (6.15)-(6.17) to system (6.1).

Proof *Let the optimization problem (6.15) be feasible at timestep t. Let us denote the corresponding optimal input policies as $[u^\star_{t|t}, u^\star_{t+1|t}(\cdot), \dots, u^\star_{t+N-1|t}(\cdot)]$. Assume the MPC controller $u^\star_{t|t}$ is applied to (6.1) in closed-loop and Θ_{t+1} is updated according to (6.7). Consider a candidate policy sequence at the next timestep as:*

$$U_{t+1}(\cdot) = [u^\star_{t+1|t}(\cdot), \dots, u^\star_{t+N-1|t}(\cdot), Kx_{t+N|t+1}]. \tag{6.18}$$

We observe the following two facts: (i) from Proposition 6.3, $\Theta_{k|t+1} \subseteq \Theta_{k|t}$, for all $k \in \{t+1, t+2, \dots, t+N\}$, and (ii) from (6.16), terminal set $\mathcal{X}_N$ is robust positive invariant for all $w \in \mathbb{W}$, and for all $\theta \in \Omega$, with state feedback controller Kx. Using (i) we conclude $[u^\star_{t+1|t}(\cdot), u^\star_{t+2|t}(\cdot), \dots, u^\star_{t+N-1|t}(\cdot)]$ is an $(N-1)$ step feasible sequence at $(t+1)$ (excluding terminal condition), since at previous timestep t, it robustly satisfied all stage constraints in (6.15) for $\Theta_{k|t}$, for all $k \in \{t+1, t+2, \dots, t+N-1\}$. With this feasible sequence, $x_{t+N|t+1} \in \mathcal{X}_N$. Using (ii) we conclude, appending the $(N-1)$ step feasible sequence with $Kx_{t+N|t+1}$ ensures $x_{t+N+1|t+1} \in \mathcal{X}_N$, satisfying the terminal constraint at $(t+1)$.

6.5.2 Input to State Stability

Recall Definition 2.8. We denote the N-step robust controllable set to the terminal set $\mathcal{X}_N$ under the MPC policy (6.17) by $\mathcal{X}_0$, which is compact and contains the origin.

Definition 6.2 *(Input to State Stability [102]): Consider system (6.1) in closed-loop with the MPC controller (6.17), obtained from (6.6)-(6.7)-(6.15), given by*

$$x_{t+1} = Ax_t + Bv^\star_{t|t} + E\theta^a_t + w_t, \ \forall t \geq 0. \tag{6.19}$$

The origin is defined as Input to State Stable (ISS), with a region of attraction $\mathcal{X}_0 \subset \mathbb{R}^n$, if there exists $\mathcal{K}_\infty$ functions $\alpha_1(\cdot)$, $\alpha_2(\cdot)$, $\alpha_3(\cdot)$, a $\mathcal{K}$ function $\sigma(\cdot)$ and a function $V(\cdot,\cdot) : \mathbb{R} \times \mathcal{X}_0 \mapsto \mathbb{R}_{\geq 0}$ continuous at the origin such that,

$$\begin{aligned} &\alpha_1(\|x_t\|) \leq V(t, x) \leq \alpha_2(\|x_t\|), \ \forall x \in \mathcal{X}_0, \ \forall t \geq 0, \\ &V(t+1, x_{t+1}) - V(t, x_t) \leq -\alpha_3(\|x_t\|) + \sigma(\|w_i + E\theta^a_i\|_{\mathcal{L}_\infty}), \end{aligned}$$

where $\|\cdot\|$ denotes the Euclidean norm and signal norm $\|d_i\|_{\mathcal{L}_\infty} = \sup_{i=\{0,1,\dots,t\}} \|d_i\|$.

Theorem 6.2 *Let Assumptions 6.1-6.4 hold. Then, the optimal cost of (6.15), i.e., $J^\star_R(\cdot,\cdot)$ is an ISS Lyapunov function for closed-loop system (6.19).*

Proof *From Assumption 6.3 we know that at timestep t, $\alpha_1(\|x_t\|_2) \leq \ell(x_t, 0) \leq J_R^\star(t, x_t)$ for some $\alpha_1(\cdot) \in \mathcal{K}_\infty$. Moreover, since (6.15) is a parametric QP, $J_R^\star(t, 0) = 0$, and $\mathcal{X}_0^R$ is compact, using similar arguments as [55, Theorem 23], we know $J_R^\star(t, x_t) \leq \alpha_2(\|x_t\|_2)$ for some $\alpha_2(\cdot) \in \mathcal{K}_\infty$. Note that as opposed to [55], our $\alpha_2(\cdot)$ is not obtained via Lipschitz continuity of the value function, since in our case, $V(t, x)$ is assumed continuous only at the origin. The existence of $\alpha_2(\cdot)$ is ensured by the compactness of the constraint sets in (6.8). Now say*

$$J_R^\star(t, x_t) = \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}^\star, v_{k|t}^\star) + Q(\bar{x}_{t+N|t}^\star) = \ell(\bar{x}_{t|t}^\star, v_{t|t}^\star) + q(\bar{x}_{t+1|t}^\star), \tag{6.20}$$

where $[\bar{x}_{t|t}^\star, \ldots, \bar{x}_{t+N|t}^\star]$ is the optimal predicted nominal trajectory under the optimal nominal input sequence

$$U_t^\star(\bar{x}_t) = [u_{t|t}^\star(\bar{x}_{t|t}), \ldots, u_{t+N-1|t}^\star(\bar{x}_{t+N-1|t})]$$

applied to nominal dynamics in (6.15), and $q(\bar{x}_{t+1|t}^\star)$ provides the total cost from $(t+1)$ to $(t+N)$ under policy $U_t^\star(\bar{x}_t)$. We proved that (6.18) is a feasible policy sequence for (6.15) at timestep $(t+1)$, where $x_{t+1} = \bar{x}_{t+1}$ is obtained with (6.19). With this feasible sequence, the optimal cost of (6.15) at $(t+1)$ is bounded as

$$J_R^\star(t+1, x_{t+1}) \leq \sum_{k=t+1}^{t+N-1} \ell(\bar{x}_{k|t+1}, v_{k|t}^\star) + Q(\bar{x}_{t+N|t+1}) = q(\bar{x}_{t+1|t+1}), \tag{6.21}$$

where we have used Assumption 6.4 and the feasible nominal trajectory

$$\bar{x}_{k|t+1} = A^{k-t-1}(Ax_t + Bv_{t|t}^\star + w_t + E\theta_t^a) + \sum_{i=t+1}^{k-1} A^{k-1-i} B u_{i|t}^\star(\bar{x}_{k|t+1}),$$

for $k = \{t+2, \ldots, t+N\}$. Moreover, we know

$$\bar{x}_{t+1|t+1} = \bar{x}_{t+1|t}^\star + w_t + E\theta_t^a. \tag{6.22}$$

Combining (6.20)–(6.22) we obtain,

$$\begin{aligned} J_R^\star(t+1, x_{t+1}) - J_R^\star(t, x_t) &= q(\bar{x}_{t+1|t}^\star + w_t + E\theta_t^a) - \ell(\bar{x}_{t|t}^\star, v_{t|t}^\star) - q(\bar{x}_{t+1|t}^\star), \\ &\leq -\ell(\bar{x}_{t|t}^\star, v_{t|t}^\star) + L_q \|w_t + E\theta_t^a\|, \\ &\leq -\ell(\bar{x}_{t|t}^\star, 0) + L_q \|w_t + E\theta_t^a\|, \\ &\leq -\alpha_3(\|x_t\|_2) + L_q \|w_i + E\theta_i^a\|_{\mathcal{L}_\infty}, \ \textit{with } \alpha_1(\cdot) = \alpha_3(\cdot), \end{aligned}$$

where $q(\cdot)$ is L_q-Lipschitz, as $q(\cdot)$ is a sum of quadratic terms in compact (6.8a). Hence the origin of (6.19) is ISS.

6.6 Numerical Simulations

We consider the following infinite horizon optimal control problem with unknown offset θ_t^a that satisfies Assumption 6.1 and Assumption-6.2:

$$\begin{aligned}
\min_{u_0,u_1(\cdot),\dots} \; & \sum_{t\geq 0} \|\bar{x}_t\|_2^2 + 10\,\|\bar{u}_t\|_2^2 \\
\text{s.t.}, \quad & x_{t+1} = Ax_t + Bu_t(x_t) + E\theta_t^a + w_t, \\
& \bar{x}_{t+1} = A\bar{x}_t + B\bar{u}_t + E\bar{\theta}_t, \\
& \begin{bmatrix} -5 \\ -2.5 \end{bmatrix} \leq x_t \leq \begin{bmatrix} 5 \\ 2.5 \end{bmatrix}, \\
& -4 \leq u_t(x_t) \leq 4, \\
& \forall w_t \in \mathbb{W},\ \forall \theta_t \in \Theta_t, \\
& x_0 = x_S,\ \bar{x}_0 = x_S, \\
& t = 0, 1, \dots,
\end{aligned} \tag{6.23}$$

where

$$A = \begin{bmatrix} 1.2 & 1.5 \\ 0 & 1.3 \end{bmatrix},\ B = [0, 1]^\top,$$

and Feasible Parameter Set Θ_t is updated based on (6.6)–(6.7) for all timesteps $t \geq 0$. The disturbance $w_t \in \mathbb{W} = \{w \in \mathbb{R}^2 : ||w||_\infty \leq 0.1\}$. The initial Feasible Parameter Set is defined as

$$\Omega = \Theta_0 = \{\theta \in \mathbb{R}^2 : [-0.5, -0.5]^\top \leq \theta \leq [0.5, 0.5]^\top\}.$$

The true offset parameter θ_t^a is time-varying, with rate bounded by the polytope

$$\mathcal{P} := [-0.05, -0.05]^\top \leq \Delta\theta_t^a \leq [0.05, 0.05]^\top.$$

For numerical simulations, we generate a true offset that starts from

$$\theta_0^a = [0.49, 0.49]^\top,$$

and has a rate of change

$$\Delta\theta_t^a = [-0.0395, -0.0395]^\top,$$

as shown in Fig. 6.1. The matrix $E \in \mathbb{R}^{2\times 2}$ is picked as the identity matrix. The robust adaptive MPC in (6.15)-(6.17) is implemented with a control horizon of $N = 6$, and the feedback gain K in (6.16) is chosen to be the optimal LQR gain for system $x^+ = Ax+Bu$ with parameters $Q_{\text{LQR}} = I_2$ and $R_{\text{LQR}} = 10$. The initial state is chosen as $x_S = [-3.21, -0.25]^\top$.

Fig. 6.1 shows the recursive adaptation of the Feasible Parameter Set and time evolution of the true offset θ_t^a. The true parameter lies within Ω, and is always captured by Θ_t at all timesteps. Fig. 6.2 shows the Monte Carlo simulations for 100 different sampled trajectories with our robust adaptive MPC, which highlights satisfaction of constraints in (6.23) robustly for all feasible offset uncertainties $\theta_t \in \Theta_t$ and disturbances $w_t \in \mathbb{W}$, for all $t \geq 0$. Such robust satisfaction of constraints is crucial for safety critical applications with an uncertain system.

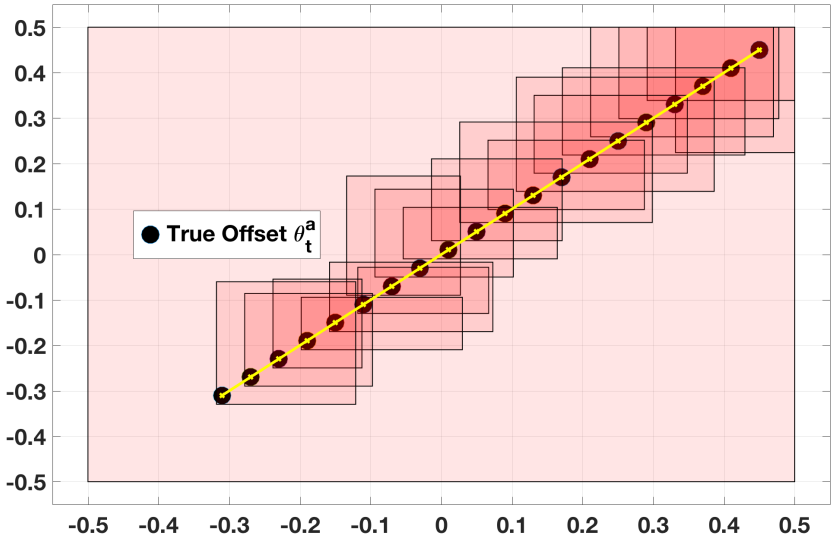


Figure 6.1: Feasible Parameter Set evolution

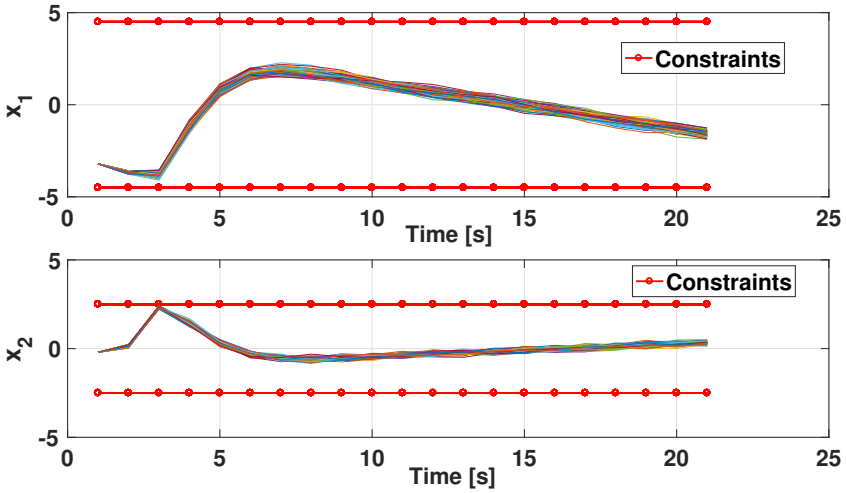


Figure 6.2: Monte Carlo simulations depicting robust constraint satisfaction

6.7 The Extension to Parametric Uncertainty in an LPV Model

Recall (2.15) and (2.16) and to be consistent with the notations in this chapter we denote $\theta_t^{\text{tr}} = \theta_t^a$. That is, we consider a specific case of model (M2) in Chapter 2 of the form:

$$x_{t+1} = A(\theta_t^a)x_t + B(\theta_t^a)u_t + w_t, \;\; w_t \in \mathbb{W},$$

where $\theta_t^a \in \mathbb{R}^p$ is a time-varying parameter unknown to the control designer, which decides the values of the system matrices as:

$$(A(\theta_t^a), B(\theta_t^a)) = (A_0, B_0) + \sum_{i=1}^{p} \theta_{i,t}^a (A_i, B_i).$$

These LPV models for robust adaptive MPC synthesis are considered in works such as [12, 13, 103]. Matrices (A_i, B_i) for $i = 0, 1, \ldots, p$ are known. Parameter $\theta_t^a \in \Omega$, with $\Delta\theta_t^a \in \mathcal{P}$ for all $t \geq 0$. An algorithm similar to the one in Section 6.3 is presented in [12] for adaptation of the feasible parameter sets Θ_t with collected data. A robust MPC algorithm presented in Chapter 3 or Chapter 4 can then be used for control design. All the guarantees of recursive feasibility and input to state stability follow.

6.7.1 ROA Enlargement

For the sake of simplicity and WLOG, consider a system defined by the following with a time-invariant parameter, i.e., $\theta_t^a = \theta^a$ for all $t \geq 0$:

$$\bar{A} = \begin{bmatrix} 0.5 & 0.2 \\ -0.1 & 0.6 \end{bmatrix}, \ \bar{B} = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix},$$
$$\theta^a = \begin{bmatrix} 0.2 & 0.8 & 0.05 \end{bmatrix}^\top, \ \Theta_0 = \{\theta : \begin{bmatrix} -1.5 \\ -1.5 \\ -1.5 \end{bmatrix} \leq \theta \leq \begin{bmatrix} 1.5 \\ 1.5 \\ 1.5 \end{bmatrix}\},$$

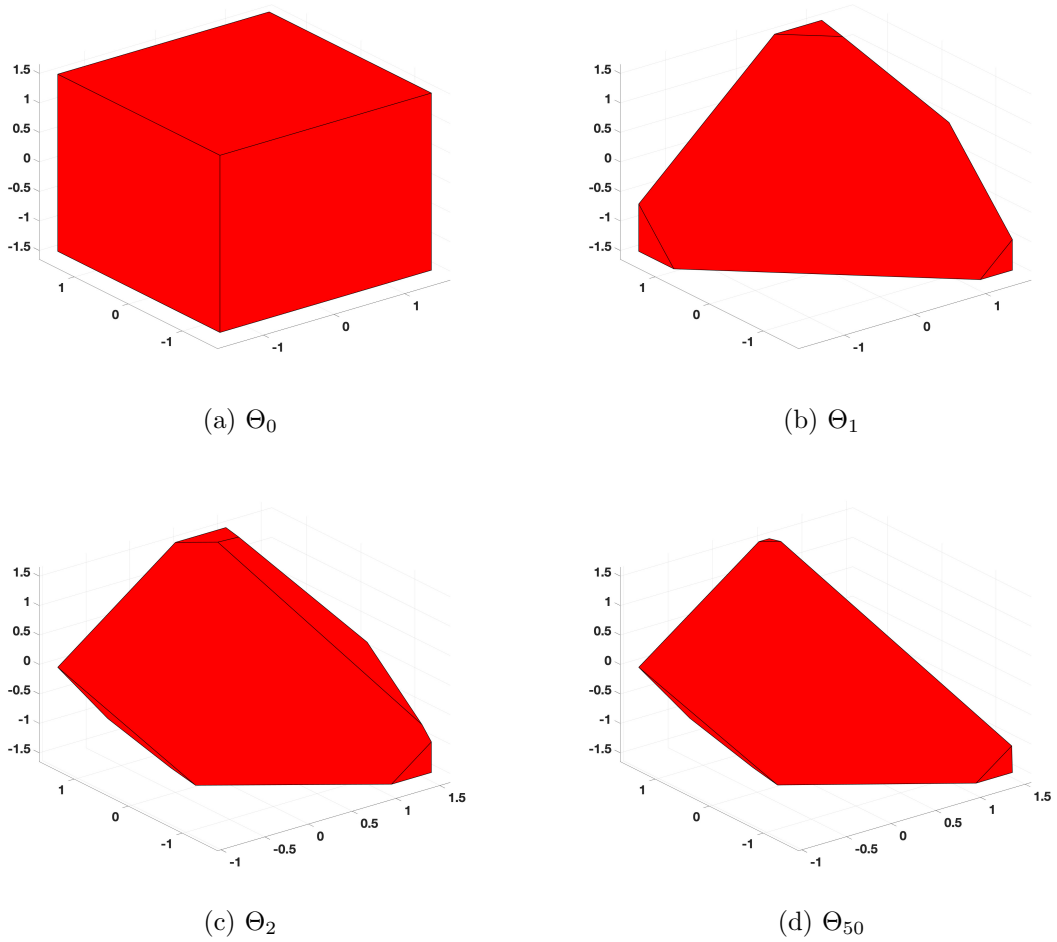
with matrices

$$A_1 = \begin{bmatrix} 0.042 & 0 \\ 0.072 & 0.03 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0.01 \\ 0 \end{bmatrix},$$
$$A_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.02 \\ 0.01 \end{bmatrix},$$
$$A_3 = \begin{bmatrix} 0.015 & 0.019 \\ 0.009 & 0.035 \end{bmatrix}, \ B_3 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

subject to box state and input constraints

$$\mathcal{X} = \{x : \begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq x \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix}\},$$
$$\mathcal{U} = \{x : -1 \leq x \leq 1\},$$

and a disturbance $\|w\|_\infty \leq 0.2$. We use the parameter adaptation algorithm of [12, Section 3.1] and the MPC algorithm of Chapter 3 with $N = 3$, $Q = 10$, $R = 1$ and a feedback gain K satisfying Assumption 3.1 chosen as $K = -[0.85, 1.5]$.



(a) Θ_0 (b) Θ_1

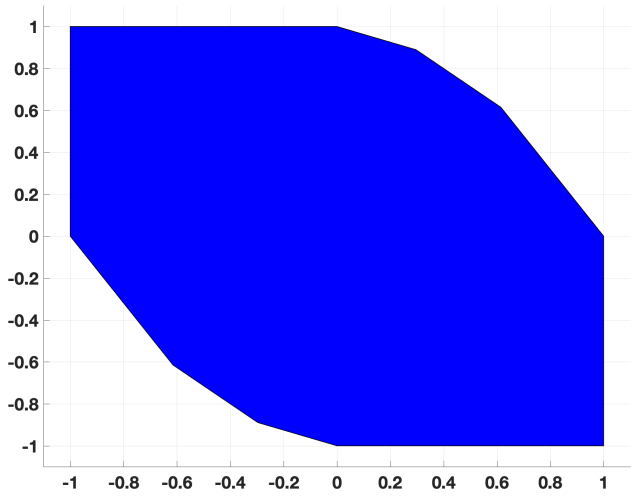
(c) Θ_2 (d) Θ_{50}

Figure 6.3: Evolution of the Feasible Parameter Sets. The Feasible Parameter Sets are shrunk and refined over time with collected state-input data from the system.

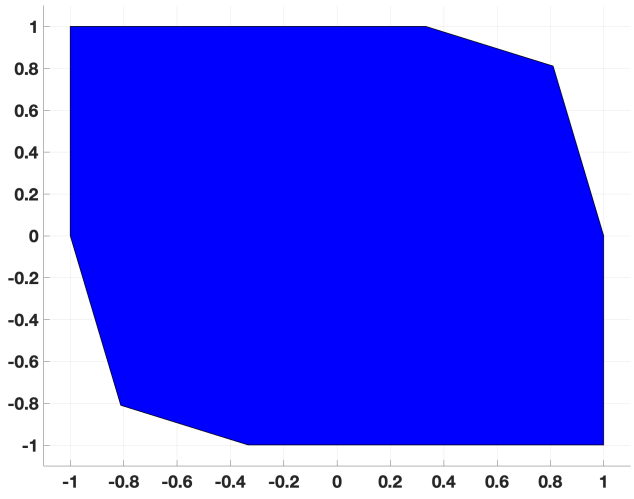
From Fig. 6.3 we see parameter adaptation altering the volume of the Feasible Parameter Sets as time passes. The Feasible Parameter Set Θ_0 is the largest in volume and is shrunk at each timestep. The effect of this lowering in the domain of model uncertainty is seen in terms of the controller properties in Fig. 6.4, where we see an increase in the volume of the ROA. The enlargement of the ROA seen in Fig. 6.4 indicates that the robust adaptive MPC lowers its conservatism due to the refinement of the Feasible Parameter Sets over time.

6.7.2 Control Computation Times

Table 6.1 further shows the computational efficiency of the algorithm. The adaptive robust MPC obtains similar online control computation times compared to the robust MPC of Chapter 3. Thus, the ROA results from Fig. 6.4 and the computation time values from



(a) Initial approx. ROA



(b) Final approx. ROA

Figure 6.4: Evolution of the approx. ROA. The approx. ROA grows over time, as the shrinking of the Feasible Parameter Sets Θ_t over time reduces the conservatism of the controller.

Table 6.1 suggests the potential of utilizing tools from Chapter 3 and Chapter 4 for adaptive MPC applications and obtaining an edge in balancing the computational complexity vs conservatism trade-off over counterpart algorithms such as [12, 13, 103, 14].

Table 6.1: Avg. online computation times [sec] using the robust MPC from Chapter 3. Values are obtained with a MacBook Pro 16inch, 2019, 2.3 GHz 8-Core Intel Core i9, 16 GB memory, using the Gurobi solver.

Horizon	**Robust Adaptive MPC for LPV Systems**
$N_t = 1$	0.0033
$N_t = 2$	0.0026
$N_t = 3$	0.0038

6.8 Chapter Summary

We first proposed a robust adaptive MPC framework for uncertain linear time-invariant systems, where we learn a bounded and time-varying additive offset-parameter uncertainty in the model with available data. We proved recursive feasibility and input to state stability of the resulting MPC algorithm. We then extended the parameter adaptation strategy to linear parameter varying systems and designed a robust adaptive MPC utilizing the control approach from Chapter 3. We demonstrated that model parameter adaptation lowers the controller conservatism over time, while keeping the control synthesis computationally efficient.

Appendix

Proof of Proposition 6.1

We prove Proposition 6.1 using induction, following the proof of the same in [97]. At timestep $t = 0$ we know that $\Theta_0 = \Omega$ and from Assumption 6.2, Ω is nonempty and $\theta_0^a \in \Omega$. Now using inductive argument, let us assume that the claim holds true for some $t \geq 0$. That is, for some nonempty Θ_t, we have $\theta_t^a \in \Theta_t$. Now we must prove $\Theta_{t+1} \neq \phi$ and $\theta_{t+1}^a \in \Theta_{t+1}$. Let us define the following matrices:

$$\begin{aligned}
\mathcal{H}_t^\theta &= [(\mathcal{H}_0^\theta)^\top, (\bar{\mathcal{H}}_t^\theta)^\top]^\top \in \mathbb{R}^{r_t \times p}, \\
h_t^\theta &= [(h_0^\theta)^\top, (\bar{h}_t^\theta)^\top]^\top \in \mathbb{R}^{r_t}, \\
\Delta h_t^\theta &= [(\mathbf{0}_{r_0})^\top, (\Delta \bar{h}_t^\theta)^\top]^\top \in \mathbb{R}^{r_t},
\end{aligned}$$

where $r_t = r_0 + 2t,\ \forall t \geq 0$ is the number of faces of the Feasible Parameter Set polytope Θ_t. Now from Assumption 6.2 we know:

$$\mathcal{H}_0^\theta \theta_{t+1}^a \leq h_0^\theta, \tag{6.24}$$

and from inductive assumptions we know that $\bar{\mathcal{H}}_t^\theta \theta_t^a \leq \bar{h}_t^\theta$. Therefore, we can ensure the following holds:

$$\bar{\mathcal{H}}_t^\theta \theta_{t+1}^a \leq \bar{h}_t^\theta + \Delta \bar{h}_t^\theta. \tag{6.25}$$

Moreover, we know that:

$$-E\theta_{t+1}^a \leq -x_{t+1} + Ax_t + Bu_t + \bar{w} - \underline{\nu}, \tag{6.26a}$$
$$+E\theta_{t+1}^a \leq x_{t+1} - Ax_t - Bu_t + \bar{w} + \bar{\nu}. \tag{6.26b}$$

Hence, from (6.24), (6.25) and (6.26) we can have, $\mathcal{H}_{t+1}^\theta \theta_{t+1}^a \leq h_{t+1}^\theta$, where

$$\mathcal{H}_{t+1}^\theta = [(\mathcal{H}_0^\theta)^\top, (\bar{\mathcal{H}}_t^\theta)^\top, -E^\top, E^\top]^\top \in \mathbb{R}^{r_{t+1} \times p}, h_{t+1}^\theta = \begin{bmatrix} h_0^\theta \\ \bar{h}_t^\theta + \Delta \bar{h}_t^\theta \\ -x_{t+1} + Ax_t + Bu_t + \bar{w} - \underline{\nu} \\ x_{t+1} - Ax_t - Bu_t + \bar{w} + \bar{\nu} \end{bmatrix} \in \mathbb{R}^{r_{t+1}}.$$

This proves that Θ_{t+1} is nonempty and contains the actual offset uncertainty θ_{t+1}^a at the $(t+1)$-th timestep. This concludes the proof.

Proof of Proposition 6.2

Utilizing the contraction property of Euclidean projection in (6.10b) similar to [92], we can write

$$\frac{1}{\mu}\|\bar{\theta}_{t+1} - \theta_{t+1}^a\|^2 - \frac{1}{\mu}\|\bar{\theta}_t - \theta_t^a\|^2 \leq \frac{1}{\mu}\|\tilde{\theta}_{t+1} - \theta_{t+1}^a\|^2 - \frac{1}{\mu}\|\bar{\theta}_t - \theta_t^a\|^2,$$

where $\|\cdot\|$ is the Euclidean norm. This gives

$$\begin{aligned}
&\frac{1}{\mu}\|\bar{\theta}_{t+1}-\theta^a_{t+1}\|^2-\frac{1}{\mu}\|\bar{\theta}_t-\theta^a_t\|^2\\
&\leq \frac{1}{\mu}\|\tilde{\theta}_{t+1}-\theta^a_t\|^2+\frac{2}{\mu}(\tilde{\theta}_{t+1}-\bar{\theta}_t)^\top(\bar{\theta}_t-\theta^a_t)+\frac{2}{\mu}\tilde{\theta}^\top_{t+1}(\theta^a_t-\theta^a_{t+1}),\\
&= \frac{1}{\mu}\|\mu E^\top(\tilde{x}_{t+1|t}+w_t)\|^2+2(\tilde{x}_{t+1|t}+w_t)^\top E(\bar{\theta}_t-\theta^a_t)+\frac{2}{\mu}\tilde{\theta}^\top_{t+1}(\theta^a_t-\theta^a_{t+1}),\\
&\leq \frac{1}{\mu}\|\mu E^\top(\tilde{x}_{t+1|t}+w_t)\|^2+2(\tilde{x}_{t+1|t}+w_t)^\top E(\bar{\theta}_t-\theta^a_t)+\frac{2}{\mu}\|\tilde{\theta}_{t+1}\|\|(\theta^a_t-\theta^a_{t+1})\|.
\end{aligned}$$

Consider Ω and $\mathcal{P}$ sets from Assumption 6.1 and Assumption 6.2. Define $\sup_{\omega\in\Omega}\|\omega\|=\omega_M$ and $\sup_{p\in\mathcal{P}}\|p\|=p_M$. Then the above inequality can be written as

$$\begin{aligned}
\frac{1}{\mu}(\|\bar{\theta}_{t+1}-\theta^a_{t+1}\|^2-\|\bar{\theta}_t-\theta^a_t\|^2) &\leq \frac{1}{\mu}\|\mu E^\top(\tilde{x}_{t+1|t}+w_t)\|^2+2(\tilde{x}_{t+1|t}+w_t)^\top E(\bar{\theta}_t-\theta^a_t)+\frac{2\omega_M p_M}{\mu},\\
&\leq (\mu\|E\|^2-1)\|\tilde{x}_{t+1|t}+w_t\|^2-\|\tilde{x}_{t+1|t}\|^2+\|w_t\|^2+\frac{2}{\mu}\omega_M p_M,\\
&\leq -\|\tilde{x}_{t+1|t}\|^2+\|w_t\|^2,
\end{aligned}$$

since from Remark 6.1 we know $\frac{1}{\mu}>\|E\|^2$, and we have used $x_{t+1}-\bar{x}_{t+1|t}=\tilde{x}_{t+1|t}+w_t$ and $\tilde{x}_{t+1|t}=E(\theta^a_t-\bar{\theta}_t)$. Summing both sides of the inequality from 0 to $\tilde{m}$ leads to a telescopic sum on the LHS, and we obtain,

$$\frac{1}{\mu}\|\bar{\theta}_{\tilde{m}+1}-\theta^a_{\tilde{m}+1}\|^2+\sum_{t=0}^{\tilde{m}}\|\tilde{x}_{t+1|t}\|^2\leq\sum_{t=0}^{\tilde{m}}\|w_t\|^2+\frac{1}{\mu}\|\bar{\theta}_0-\theta^a_0\|^2,$$

which, upon division by RHS on both sides gives

$$\sup_{\tilde{m}\in\mathbb{N},w_t\in\mathbb{W},\bar{\theta}_0\in\Omega}\frac{\sum_{t=0}^{\tilde{m}}\|\tilde{x}_{t+1|t}\|^2}{\frac{1}{\mu}\|\bar{\theta}_0-\theta^a_0\|^2+\sum_{t=0}^{m}\|w_t\|^2}\leq 1.$$

Proof of Proposition 6.3

From the definition of $\Theta_{t+1|t}$ in (6.12), we see that,

$$\mathcal{H}^\theta_{t+1}=[(\mathcal{H}^\theta_{t+1|t})^\top,-E^\top,+E^\top]^\top,\ h^\theta_{t+1}=\begin{bmatrix}h^\theta_{t+1|t}\\-x_{t+1}+Ax_t+Bu_t+\bar{w}-\underline{\nu}\\x_{t+1}-Ax_t-Bu_t+\bar{w}+\bar{\nu}\end{bmatrix}.$$

So $\Theta_{t+1|t+1} \subseteq \Theta_{t+1|t}$. Now, the matrices of the Predicted Feasible Parameter Sets at next timestep, $\mathcal{H}^\theta_{k|t+1}$ and $h^\theta_{k|t+1}$ for all $k \in \{t+2, \dots, t+N-1\}$ are formed from $\mathcal{H}^\theta_{t+1}$ and h^θ_{t+1} by construction. Therefore, for all $k \in \{t+2, \dots, t+N-1\}$,

$$\mathcal{H}^\theta_{k|t+1} = [(\mathcal{H}^\theta_{k|t})^\top, -E^\top, +E^\top]^\top,\ h^\theta_{k|t+1} = \begin{bmatrix} h^\theta_{k|t} \\ -x_{t+1} + Ax_t + Bu_t + \bar{w} - \underline{\nu} \\ x_{t+1} - Ax_t - Bu_t + \bar{w} + \bar{\nu} \end{bmatrix},$$

where $\mathcal{H}^\theta_{k|t}$ and $h^\theta_{k|t}$ are given by (6.12). So, for all $k \in \{t+2, \dots, t+N-1\}$, each of the sets for $\Theta_{k|t+1}$ are formed by the same inequalities which form $\Theta_{k|t}$, appended by two extra rows from the new measurement. Therefore, $\Theta_{k|t+1} \subseteq \Theta_{k|t}$ for all $k \in \{t+2, \dots, t+N-1\}$. Moreover, from (6.13), $\Theta_{t+N|t} = \Omega$. Using this,

$$\mathcal{H}^\theta_{t+N|t+1} = [(\mathcal{H}^\theta_{0})^\top, -E^\top, +E^\top]^\top,\ h^\theta_{t+N|t+1} = \begin{bmatrix} h^\theta_{0} \\ -x_{t+1} + Ax_t + Bu_t + \bar{w} - \underline{\nu} \\ x_{t+1} - Ax_t - Bu_t + \bar{w} + \bar{\nu} \end{bmatrix},$$

and therefore $\Theta_{t+N|t+1} \subseteq \Theta_{t+N|t} = \Omega$ from the definition of Ω in (6.4).

Dualization of the MPC Problem

In this section we show how the robust MPC problem (6.15) can be reformulated for efficient solving. The constraints in (6.15) can be compactly written with similar notations as [55]:

$$F_R \mathbf{v}_t + \max_{\mathbf{w}_t, \boldsymbol{\theta}_t} (F_R \mathbf{M}_t + G_R)(\mathbf{w}_t + \mathbf{E}\boldsymbol{\theta}_t) \leq c_R + H_R x_t, \tag{6.27}$$

where we denote, $\mathbf{v}_t = [v^\top_{t|t}, v^\top_{t+1|t}, \dots, v^\top_{t+N-1|t}]^\top \in \mathbb{R}^{mN}$, $\boldsymbol{\theta}_t = [\theta^\top_{t|t}, \dots, \theta^\top_{t+N-1|t}]^\top \in \mathbb{R}^{pN}$ for all $\theta_{k|t} \in \Theta_{k|t}$, for all $k \in \{t, \dots, t+N-1\}$, $\mathbf{E} = \mathrm{diag}(E, \dots, E) \in \mathbb{R}^{nN \times pN}$ and $\mathbf{w}_t = [w^\top_{t|t}, \dots, w^\top_{t+N-1|t}]^\top \in \mathbb{R}^{nN}$. The matrices above in (6.27) $F_R \in \mathbb{R}^{(sN+r_R)\times mN}, G_R \in \mathbb{R}^{(sN+r_R)\times nN}, c_R \in \mathbb{R}^{sN+r_R}$ and $H_R \in \mathbb{R}^{(sN+r_R)\times n}$ are obtained as:

$$F_R = \begin{bmatrix} D & \mathbf{0}_{s\times m} & \cdots & \mathbf{0}_{s\times m} \\ CB & D & \cdots & \mathbf{0}_{s\times m} \\ \vdots & \ddots & \ddots & \vdots \\ CA^{N-2}B & CA^{N-3}B & \cdots & D \\ Y_R A^{N-1}B & Y_R A^{N-2}B & \cdots & Y_R B \end{bmatrix},\ G_R = \begin{bmatrix} \mathbf{0}_{s\times n} & \mathbf{0}_{s\times n} & \cdots & \mathbf{0}_{s\times n} \\ C & \mathbf{0}_{s\times n} & \cdots & \mathbf{0}_{s\times n} \\ \vdots & \ddots & \ddots & \vdots \\ CA^{N-2} & CA^{N-3} & \cdots & \mathbf{0}_{s\times n} \\ Y_R A^{N-1} & Y_R A^{N-2} & \cdots & Y_R \end{bmatrix},$$

$$c_R = [b^\top, \dots, b^\top, z_R^\top],$$

$$H_R = -[C^\top, (CA)^\top, \dots, (CA^{N-1})^\top, (Y_R A^N)^\top]^\top.$$

For $k = \{t, \dots, t+N-1\}$, denote the set of polytopes $\mathbb{S}^R_{k|t} = \{w \in \mathbb{W},\ \theta \in \Theta_{k|t} : S^R_{k|t}(w + E\theta) \leq h^R_{k|t}\}$. Then we can define a polytope $\mathbb{S}_R = \{\mathbf{w}_t + \mathbf{E}\boldsymbol{\theta}_t \in \mathbb{R}^{nN} : S^R(\mathbf{w}_t + \mathbf{E}\boldsymbol{\theta}_t) \leq h^R, S^R \in$

$\mathbb{R}^{a_R \times nN}$, $h^R \in \mathbb{R}^{a_R}\}$ with, $S^R = \text{diag}(S^R_{t|t}, \ldots, S^R_{t+N-1|t})$, $h^R = [(h^R_{t|t})^\top, \ldots, (h^R_{t+N-1|t})^\top]^\top$. Now (6.27) can be written with auxiliary decision variables $Z_R \in \mathbb{R}^{a_R \times (sN+r_R)}$ using *duality* of linear programs as,

$$\begin{aligned} &F_R \mathbf{v}_t + Z_R^\top h^R \leq c_R + H_R x_t, \\ &(F_R \mathbf{M}_t + G_R) = Z_R^\top S^R, \\ &Z_R \geq 0, \end{aligned}$$

which is a tractable linear programming problem that can be efficiently solved with any existing solver for real-time implementation of Algorithm 4.

Chapter 7

Learning Disturbance Distribution Supports in Robust MPC

This chapter is based on the published work [36]. In this chapter, we present an approach to design an MPC controller for systems of the form (M1) in Chapter 2, performing an iterative task [104], where the support of the additive disturbance is learned from collected iteration data. As the true support is unknown, robust satisfaction of the imposed constraints are not guaranteed by the resultant MPC which uses the learned estimates. Instead, we guarantee a user-specified upper bound on the probability of constraint violations by the closed-loop system over all iterations.

7.1 Summary of Contributions

The main contributions of this chapter can be summarized as:

- We introduce the notion of a *Confidence Support*, which is guaranteed to contain the true disturbance support with a specified probability. Constructing and updating the Confidence Supports after each iteration is computationally cheap, unlike [34].
- Using these Confidence Supports, we present a robust MPC design and demonstrate satisfaction of desired upper bound on probability of failure in each iteration. For any value of user-specified upper bound on probability of failure, the controller is able to learn robust satisfaction of imposed constraints asymptotically, without suffering conservatism that is inherent to existing approaches [22, 92, 13].

7.2 Problem Formulation

We consider uncertain linear time-invariant systems of the form (2.9), with $x_t \in \mathbb{R}^d$ as the state at timestep t, $u_t \in \mathbb{R}^m$ the input, and A and B are known system matrices of

appropriate dimensions[1]. At each timestep t, the system is affected by an independently and identically distributed (i.i.d.) random disturbance $w_t \overset{\text{iid}}{\sim} \mathcal{P}$ with a convex and compact support $\mathbb{W} \subset \mathbb{R}^d$. We aim to satisfy state and input constraints on the system robustly. We define $H_x \in \mathbb{R}^{s\times d}$, $h_x \in \mathbb{R}^s$, $H_u \in \mathbb{R}^{o\times m}$ and $h_u \in \mathbb{R}^o$. We can then write the imposed constraints $\mathcal{X}$ and $\mathcal{U}$ for all timesteps $t \geq 0$, given by:

$$\mathcal{X} = \{x \in \mathbb{R}^d \mid H_x x \leq h_x\},\ \mathcal{U} = \{u \in \mathbb{R}^m \mid H_u u \leq h_u\}$$

jointly as:

$$\mathbb{Z} := \{(x,u) : H_x x \leq h_x,\ H_u u \leq h_u\}. \tag{7.2}$$

We assume that system (2.9) performs the same task repeatedly for J number of times. Each task execution is referred to as *iteration*. Our goal is to design a controller that, at each iteration j, solves the finite horizon robust optimal control problem:

$$\begin{aligned}
\min_{u_0^j, u_1^j(\cdot), \ldots} \quad & \sum_{t=0}^{T-1} \ell\left(\bar{x}_t^j, u_t^j\left(\bar{x}_t^j\right)\right) \\
\text{s.t.,} \quad & x_{t+1}^j = Ax_t^j + Bu_t^j(x_t^j) + w_t^j, \\
& \bar{x}_{t+1}^j = A\bar{x}_t^j + Bu_t^j(\bar{x}_t^j), \\
& H_x x_t^j \leq h_x,\ H_u u_t^j(x_t^j) \leq h_u, \\
& \forall w_t^j \in \mathbb{W}, \\
& x_0^j = x_S,\ t = \{0, 1, \ldots, (T-1)\},
\end{aligned} \tag{7.3}$$

where x_t^j, u_t^j and w_t^j denote the realized system state, control input and disturbance at timestep t of the j^{th} iteration respectively, and $(\bar{x}_t^j, u_t^j(\bar{x}_t^j))$ denote the disturbance-free nominal state and corresponding nominal input. Notice that (7.3) minimizes the nominal cost over a time horizon of length $T \gg 0$ in any j^{th} iteration with $j \in [J]$. Here we use $[J]$ to denote the set $\{1, 2, \ldots, J\}$. As task duration $T \gg 0$, for computational tractability we try to approximate a solution to the optimal control problem (7.3), by solving a simpler constrained optimal control problem with prediction horizon $N \ll T$ in a receding horizon fashion.

In this chapter, we consider the support $\mathbb{W}$ of disturbance w_t^j to be an unknown, convex and compact set. We estimate $\mathbb{W}$ using observed disturbance samples. At the start of iteration j, the estimated support is $\hat{\mathbb{W}}^j$.

7.3 Iterative MPC Problem

The MPC controller solves a finite horizon optimal control problem at each timestep t in the j^{th} iteration. Since the disturbance support $\mathbb{W}$ is unknown and is estimated with $\hat{\mathbb{W}}^j$ built

[1]Note the use of d in this chapter for the dimension of the state-space, as n will be used to denote the number of data samples.

from data, robust satisfaction of (7.2) along the iteration is not guaranteed. This implies that the closed-loop task execution might fail. We will formally define this notion of *failure* after defining the closed-loop controller in this section. We attempt to design a robust MPC controller in the j^{th} iteration with our best estimate $\hat{\mathbb{W}}^j$ of disturbance support $\mathbb{W}$, by solving the following optimal control problem:

$$
\begin{aligned}
&V^{\text{MPC},j}_{t\to t+N}(x^j_t, \hat{\mathbb{W}}^j, \hat{\mathcal{X}}^j_N) := \\
&\quad \min_{U^j_t(\cdot)} \quad \sum_{k=t}^{t+N-1} \ell(\bar{x}^j_{k|t}, v^j_{k|t}) + Q(\bar{x}^j_{t+N|t}) \\
&\quad \text{s.t.,} \quad x^j_{k+1|t} = Ax^j_{k|t} + Bu^j_{k|t} + w^j_{k|t}, \\
&\qquad\qquad \bar{x}^j_{k+1|t} = A\bar{x}^j_{k|t} + Bv^j_{k|t}, \\
&\qquad\qquad u^j_{k|t} = \sum_{l=t}^{k-1} M^j_{k,l|t} w^j_{l|t} + v^j_{k|t}, \\
&\qquad\qquad H_x x^j_{k|t} \le h_x,\ H_u u^j_{k|t} \le h_u, \\
&\qquad\qquad x^j_{t+N|t} \in \hat{\mathcal{X}}^j_N, \\
&\qquad\qquad \forall w^j_{k|t} \in \hat{\mathbb{W}}^j, \\
&\qquad\qquad \forall k = \{t, \dots, t+N-1\},\ x^j_{t|t} = \bar{x}^j_{t|t} = x^j_t,
\end{aligned}
\tag{7.4}
$$

where $U^j_t(\cdot) = \{u^j_{t|t}, \dots, u^j_{t+N-1|t}(x_{t+N-1|t})\}$, x^j_t is the measured state at timestep t, $x^j_{k|t}$ is the prediction of state at timestep k, obtained by applying predicted input policies $\{u^j_{t|t}, \dots, u^j_{k-1|t}(x_{k-1|t})]$ to system (2.9) and $\{\bar{x}^j_{k|t}, v^j_{k|t}\}$ with $v^j_{k|t} = u^j_{k|t}(\bar{x}^j_{k|t})$ denote the disturbance-free nominal state and corresponding input respectively. In (7.4) we have used the affine disturbance feedback parametrization to tackle challenge (C2) introduced in Chapter 2. The MPC controller minimizes the cost over the predicted disturbance free nominal trajectory $\left\{\{\bar{x}^j_{k|t}, v^j_{k|t}\}_{k=t}^{t+N-1}, \bar{x}^j_{t+N|t}\right\}$, which comprises of the positive definite stage cost $\ell(\cdot,\cdot)$, and the terminal cost $Q(\cdot)$. We use state feedback to construct terminal set $\hat{\mathcal{X}}^j_N = \{x \in \mathbb{R}^d : \hat{Y}^j x \le \hat{z}^j,\ \hat{Y}^j \in \mathbb{R}^{r^j \times d},\ \hat{z}^j \in \mathbb{R}^{r^j}\}$, which is the $(T-N)$ step robust controllable set to the set of state constraints in (7.2), obtained with a state feedback controller $u = Kx$, dynamics (2.9) and constraints (7.2). This set has the properties:

$$
\begin{aligned}
&\hat{\mathcal{X}}^j_N \subseteq \{x | (x, Kx) \in \mathbb{Z}\}, \\
&H_x((A+BK)^i x + \sum_{\tilde{i}=0}^{i-1} (A+BK)^{i-\tilde{i}-1} w_{\tilde{i}}) \le h_x, \\
&H_u(K((A+BK)^i x + \sum_{\tilde{i}=0}^{i-1} (A+BK)^{i-\tilde{i}-1} w_{\tilde{i}})) \le h_u, \\
&\forall x \in \hat{\mathcal{X}}^j_N,\ \forall w_i \in \hat{\mathbb{W}}^j,\ \forall i = 1, 2, \dots, (T-N).
\end{aligned}
\tag{7.5}
$$

After solving (7.4), in closed-loop, we apply

$$u_t^j = v_{t|t}^{j,\star} \tag{7.6}$$

to system (2.9). We then resolve the problem (7.4) again at the next $(t+1)$-th timestep, yielding a receding horizon strategy.

Remark 7.1 *Computing sets such as (7.5) can become expensive in certain scenarios, where for example the number of constraints in $\mathbb{Z}$, or the dimension d of states is too large. In such cases one may opt for data driven methods such as [104, 105] or simple approximation methods such as [106, 107] to construct these terminal sets.*

Assumption 7.1 (Well Posedness) *We assume that given an initial state x_S, optimization problem (7.4) is feasible at all timesteps $0 \leq t \leq T-1$ with true uncertainty support $\hat{\mathbb{W}}^j = \mathbb{W}$ for all iterations $j \in [J]$.*

Since $\mathbb{W}$ is unknown and is being estimated with $\hat{\mathbb{W}}^j$ in the j^{th} iteration, we might lose the feasibility of (7.4) during $0 \leq t \leq T-1$. We formalize this with the following definition:

Definition 7.1 (State Constraint Failure) *A State Constraint Failure at timestep t in iteration j is the event*

$$[\text{SCF}]_t^j : \; H_x x_t^j > h_x. \tag{7.7}$$

That is, a State Constraint Failure implies the violation of imposed constraints (7.2) by system (2.9) in closed-loop with MPC controller (7.6).

Remark 7.2 *Let $T^j < T$ denote the timestep in the j^{th} iteration when a State Constraint Failure occurs. In that case, problem (7.4) becomes infeasible at T^j. We then stop the j^{th} iteration and update $\hat{\mathbb{W}}^j \overset{\text{update}}{\longrightarrow} \hat{\mathbb{W}}^{j+1}$. When $T^j = T$, it denotes a successful iteration without any State Constraint Failure.*

The probability of State Constraint Failure $[\text{SCF}]_t^j$ is a function of the sets $\hat{\mathbb{W}}^j$. In certain safety critical applications, it is necessary to keep the probability of $[\text{SCF}]_t^j$ very low, whereas in other applications a higher probability can be tolerated. However, it is not enough to focus on probability of $[\text{SCF}]_t^j$ alone. For example, a low probability of $[\text{SCF}]_t^j$ can be achieved by considering worst-case apriori estimates for $\mathbb{W}$ but it results in deteriorated controller "performance". Thus, it is desirable to not only keep probability of $[\text{SCF}]_t^j$ low, but also maintain satisfactory controller performance during successful iterations (as defined in Remark 7.2). Let the closed-loop cost of a successful iteration j be denoted by

$$\hat{\mathcal{V}}^j(x_S, w^{1:j}) = \sum_{t=0}^{T-1} \ell(x_t^j, v_{t|t}^{j,\star}). \tag{7.8}$$

where notation $w^{1:j}$ denotes the set $\bigcup_{i=1}^{j}\bigcup_{t=0}^{T-1} w_t^i$. We use average closed-loop cost $\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^{1:j})]$ to quantify controller performance. The goal is to lower the *performance loss* defined as

$$[\text{PL}]^j = |\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^{1:j})] - \mathbb{E}[\mathcal{V}^\star(x_S, w^{1:j})]|, \tag{7.9}$$

where $\mathbb{E}[\mathcal{V}^\star(x_S, w^{1:j})]$ denotes the average closed-loop cost of the j^{th} iteration if $\mathbb{W}$ had been known, i.e., $\hat{\mathbb{W}}^j = \mathbb{W}$ for all $j \in [J]$.

In the next section, we introduce two design specifications (D1) and (D2) to formalize this joint focus on lowering probability of State Constraint Failure and maintaining satisfactory controller performance. We then show how the sets $\hat{\mathbb{W}}^j$ are constructed according to these specifications.

7.4 LRBF: Learning Robustness with Bounded Failure

We consider the following design specifications:

(D1) Closed-loop MPC control law (7.6) ensures that system (2.9) in the j^{th} iteration satisfies a user specified upper bound α on probability of State Constraint Failure (Definition 7.1),

(D2) Minimize $[\text{PL}]^j$ (as defined in (7.9)) over all iterations $j \in [J]$ while satisfying (D1).

For satisfaction of (D1) we require,

$$\mathbb{P}(H_x x_t^j > h_x) \leq \alpha. \tag{7.10}$$

Since the above probability is difficult to compute, we consider an alternative notion of failure in order to upper bound the probability of State Constraint Failure.

Definition 7.2 (Disturbance Support Failure) *A Disturbance Support Failure at any timestep t in iteration j is the event*

$$[\text{DSF}]_t^j : \; w_t^j \notin \hat{\mathbb{W}}^j. \tag{7.11}$$

As the MPC controller (7.4) is robust to all $w_t^j \in \hat{\mathbb{W}}^j$, we have $[\text{SCF}]_t^j \subseteq [\text{DSF}]_t^j$. Therefore, probability of Disturbance Support Failure is an upper bound for probability of State Constraint Failure, i.e., $\mathbb{P}([\text{SCF}]_t^j) \leq \mathbb{P}([\text{DSF}]_t^j)$. Therefore, we focus on the following specification:

$$\mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j) \leq \alpha. \tag{7.12}$$

In the next few sections, we discuss how such sets $\hat{\mathbb{W}}^j$ can be constructed based on disturbance samples observed during the iterative task.

7.4.1 Need for Distributional Assumption on $\mathcal{P}$

Consider i.i.d. samples $Z_{1:n} = (Z_1, \ldots, Z_n)$ from an unknown distribution $\mathcal{P}$. All we know about the distribution is that its support $\mathbb{S}$ is convex and compact. Our objective is to find an estimate $\hat{\mathbb{S}}(Z_{1:n})$ for the support $\mathbb{S}$ such that for a user specified failure probability α,

$$\mathbb{P}(\bar{Z} \notin \hat{\mathbb{S}}(Z_{1:n})) \leq \alpha, \tag{7.13}$$

where $\bar{Z}$ is an i.i.d. draw from $\mathcal{P}$. The convex hull $C^{\text{hull}}(Z_{1:n})$ of observed samples $(Z_1, \ldots, Z_n)$ is an intuitive estimator for the support $\mathbb{S}$. It is clear that $C^{\text{hull}}(Z_{1:n}) \subseteq \mathbb{S}$. Let $\mathcal{A} \setminus \mathcal{B}$ denote the set $\{y \mid y \in \mathcal{A} \text{ and } y \notin \mathcal{B}\}$. It turns out that $\mathbb{P}(\mathbb{S} \setminus C^{\text{hull}}(Z_{1:n})) \to 0$ as $n \to \infty$ [108], i.e., $C^{\text{hull}}(Z_{1:n})$ asymptotically converges to the support $\mathbb{S}$. However, $C^{\text{hull}}(Z_{1:n})$ may not satisfy (7.13) for an arbitrary user specified failure probability α. In order to do so, $C^{\text{hull}}(Z_{1:n})$ may need to be scaled up in a suitable manner. We illustrate through a simple example that an upper bound on failure probability cannot be guaranteed without additional assumptions on the distribution $\mathcal{P}$.

Consider an unknown univariate distribution $\mathcal{P}$ with support $\mathbb{S} \subset \mathbb{R}$. Suppose we observe i.i.d. samples $Z_{1:4} = \{-1, 0.5, 1, -0.2\}$ from this distribution. The objective is to find $\hat{\mathbb{S}}(Z_{1:4})$ that satisfies (7.13) with $\alpha = 0.1$. As we know that $\mathbb{S}$ is convex and compact, it is clear that $C^{\text{hull}}(Z_{1:4}) = [-1, 1] \subseteq \mathbb{S}$. However, it is unclear whether $\hat{\mathbb{S}} = C^{\text{hull}}(Z_{1:4})$ would satisfy (7.13) with $\alpha = 0.1$. Consider two potential distributions $\mathcal{P}_1, \mathcal{P}_2$ with densities $p_1(\cdot), p_2(\cdot)$ respectively such that

$$\begin{aligned} p_1(z) &= 0.4\mathbb{I}\{|z| \leq 1\} + 0.1\mathbb{I}\{1 \leq |z| \leq 2\}, \\ p_2(z) &= 0.4\mathbb{I}\{|z| \leq 1\} + 0.01\mathbb{I}\{1 \leq |z| \leq 11\}, \end{aligned}$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function. Note that both these distributions are equally likely to generate the observed samples as they have the same distribution on $C^{\text{hull}}(Z_{1:4}) = [-1, 1]$. Observe that $\hat{\mathbb{S}} = 1.5C^{\text{hull}}(Z_{1:4})$ satisfies (7.13) for $\mathcal{P}_1$, whereas $\hat{\mathbb{S}}$ has to be set to $6C^{\text{hull}}(Z_{1:4})$ to get the same probability of failure for $\mathcal{P}_2$. Thus, without any additional assumption about the distribution, it is not possible to give any probability of failure guarantees just based on sets constructed from observed samples.

Assumption 7.2 *We assume that the unknown distribution $\mathcal{P}$ defined in Section 7.2 belongs to a finite dimensional parametric family $\{\mathcal{P}_\theta : \theta \in \Theta, \Theta \subseteq \mathbb{R}^l\}$.*

We next explore how to construct the sets $\hat{\mathbb{W}}^j$ using Assumption 7.2, so that design specification (D1) is satisfied. For that purpose, we introduce the notion of Confidence Supports which are closely related to the notion of confidence intervals in classical statistics. Subsequently in Section 7.4.3 we present our algorithm.

7.4.2 Confidence Support of a Distribution

Consider i.i.d. samples $Z_{1:n} = (Z_1, \ldots, Z_n)$ from a distribution $\mathcal{P}_\theta$ parametrized by $\theta \in \mathbb{R}$, i.e., $Z_i \overset{\text{iid}}{\sim} \mathcal{P}_\theta$. In classical statistics, the notion of confidence interval provides a convenient way to characterize the uncertainty of parameter θ from the observed samples $Z_{1:n}$.

Definition 7.3 (Confidence Interval) *A set $\mathcal{C}(Z_{1:n})$ is a $(1-\alpha)$-confidence interval for the parameter θ of distribution $\mathcal{P}_\theta$ if*

$$\mathbb{P}(\theta \notin \mathcal{C}(Z_{1:n})) \leq \alpha. \tag{7.14}$$

If $\theta \in \mathbb{R}^d,\ d > 1$, then the term *confidence region* is used for the set $\mathcal{C}(Z)$ as defined above.

Remark 7.3 *Note that $\mathcal{C}(Z)$ is a random set as it is a function of the collection of random samples $Z_{1:n}$, whereas θ is an unknown deterministic parameter. We refer the reader to [51, Chapter 9] for an introduction to confidence intervals and methods to compute them.*

We now introduce an analogous definition for the support of a distribution.

Definition 7.4 (Confidence Support) *A set $\mathcal{S}(Z_{1:n})$ is a $(1-\alpha)$-Confidence Support of a distribution $\mathcal{P}_\theta$ with support $\mathbb{S}_\theta$ if*

$$\mathbb{P}(\mathbb{S}_\theta \subseteq \mathcal{S}(Z_{1:n})) \geq 1 - \alpha, \tag{7.15}$$

i.e., $\mathcal{S}(Z_{1:n})$ contains the support $\mathbb{S}_\theta$ of $\mathcal{P}_\theta$ with probability greater than or equal to $(1-\alpha)$.

Using the above notion of Confidence Supports, we now demonstrate how the disturbance support estimates $\hat{\mathbb{W}}^j$ (as defined in iterative MPC problem (7.4)) can be computed based on observed disturbance samples.

7.4.3 Computing $\hat{\mathbb{W}}^j$

Consider i.i.d. disturbance samples $w_t^j \sim \mathcal{P}_\theta,\ \theta \in \mathbb{R}^l$ with support $\mathbb{W}$. Let $w_t^j(q)$ denote the q^{th} element of $w_t^j \in \mathbb{R}^d$. Let $w^{1:j}$ denote the set $\bigcup_{i=1}^{j} \bigcup_{t=0}^{T^i} w_t^i$. Recall that $[d]$ denotes the set $\{1, 2, \ldots, d\}$. We make the following simplifying assumption:

Assumption 7.3 *The elements of random vector $w_t^i \in \mathbb{R}^d$ are independently distributed,*

$$w_t^j(q) \sim \mathcal{P}^q_{\theta_q},\ q \in [d], \tag{7.16}$$

where $\theta = (\theta_1, \ldots, \theta_d)$ and $\{\mathcal{P}^q_{\theta_q} : \theta_q \in \Theta_q,\ \Theta_q \subset \mathbb{R}^{l/d}\}$ is the corresponding parametric family for the q^{th} element. Remark 7.4 contains a discussion about the general case.

At the start of the j^{th} iteration, the collection of samples $w^{1:j-1}$ would have been observed. As the uncertainty distribution $\mathcal{P}_\theta$ is completely specified by θ, we can compute a $(1-\alpha)$-Confidence Support $\hat{\mathbb{W}}^j\big(w^{1:j-1}\big)$ by computing confidence regions for the individual parameters $(\theta_1, \ldots, \theta_d)$. Note that the confidence regions and supports are functions of the observed disturbance samples $w^{1:j-1}$. For notational convenience, we represent such sets without explicitly showing this dependence.

Lemma 7.1 *Let $\hat{\Theta}_q^j$ be a $(1-\alpha_q)$-confidence region for θ_q. Consider $\hat{\mathbb{W}}_q^j = \bigcup_{\bar{\theta}_q \in \hat{\Theta}_q^j} \mathrm{Supp}(\mathcal{P}_{\bar{\theta}_q}^q)$, where $\mathrm{Supp}(\mathcal{P}_{\bar{\theta}_q}^q)$ denotes the support of distribution $\mathcal{P}_{\bar{\theta}_q}^q$. Then, $\hat{\mathbb{W}}^j = \hat{\mathbb{W}}_1^j \times \cdots \times \hat{\mathbb{W}}_d^j$ is a $(1 - \sum_q \alpha_q)$-Confidence Support of $\mathcal{P}_\theta$.*

Proof *By definition, $\mathbb{W} = \mathrm{Supp}(\mathcal{P}_{\theta_1}^1) \times \cdots \times \mathrm{Supp}(\mathcal{P}_{\theta_d}^d)$. As $\hat{\mathbb{W}}^j = \hat{\mathbb{W}}_1^j \times \cdots \times \hat{\mathbb{W}}_d^j$, we have*

$$\begin{aligned}
\mathbb{P}(\mathbb{W} \not\subseteq \hat{\mathbb{W}}^j) &= \mathbb{P}\Big(\bigcup_{q=1}^{d} \ \mathrm{Supp}(\mathcal{P}_{\theta_q}^q) \not\subseteq \hat{\mathbb{W}}_q^j\Big) \\
&= \mathbb{P}\Big(\bigcup_{q=1}^{d} \ \theta_q \notin \hat{\Theta}_q^j\Big), \\
&\leq \sum_{q=1}^{d} \ \mathbb{P}(\theta_q \notin \hat{\Theta}_q^j), \qquad (7.17) \\
&\leq \sum_{q=1}^{d} \ \alpha_q, \qquad (7.18)
\end{aligned}$$

where (7.17) follows from the union bound and (7.18) follows from $\hat{\Theta}_q^j$ being a $(1-\alpha_q)$-confidence region for θ_q.

Thus, a $(1-\alpha)$-Confidence Support can be constructed using $(1-\alpha_q)$-confidence regions by setting $\alpha_q = \frac{\alpha}{d}$. We now show that such a Confidence Support has a bounded probability of Disturbance Support Failure, as defined in (7.11).

Proposition 7.1 *Let $\hat{\mathbb{W}}^j$ be a $(1-\alpha)$-Confidence Support of $\mathcal{P}_\theta$ computed using samples $w^{1:j-1}$. Then, we have*

$$\mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j) \leq \alpha, \ 0 \leq t \leq T-1. \qquad (7.19)$$

Proof *Note that both w_t^j and $\hat{\mathbb{W}}^j$ are random. Using the law of total probability, we have*

$$\begin{aligned}
\mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j) &= \mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j | \mathbb{W} \subseteq \hat{\mathbb{W}}^j)\mathbb{P}(\mathbb{W} \subseteq \hat{\mathbb{W}}^j) + \mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j | \mathbb{W} \not\subseteq \hat{\mathbb{W}}^j)\mathbb{P}(\mathbb{W} \not\subseteq \hat{\mathbb{W}}^j), \\
&= \mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j | \mathbb{W} \not\subseteq \hat{\mathbb{W}}^j)\mathbb{P}(\mathbb{W} \not\subseteq \hat{\mathbb{W}}^j), \\
&\leq \mathbb{P}(\mathbb{W} \not\subseteq \hat{\mathbb{W}}^j), \qquad (7.20) \\
&\leq \alpha, \qquad (7.21)
\end{aligned}$$

where (7.21) follows from the fact that $\hat{\mathbb{W}}^j$ is a $(1-\alpha)$-Confidence Support of $\mathcal{P}_\theta$.

Remark 7.4 *The Confidence Supports constructed in this section also hold in the case that the elements of w_t^j are dependent. However, as we are not exploiting the correlations across dimensions, the above approach would yield a hyper-rectangle outer-approximation to the actual support as iteration j goes to infinity. Confidence regions for the parameter θ rather than individual elements θ_q are needed in such a case to converge to the true support, but such regions are in general difficult to compute.*

Remark 7.5 *As long as the confidence regions $\hat{\Theta}_q^j$ converge to the true parameter θ_q in probability, the Confidence Supports asymptotically converge to the true uncertainty support, i.e., $\hat{\mathbb{W}}^j \to \mathbb{W}$ in probability. The MPC controller (7.6) thus asymptotically learns to satisfy (7.2) robustly.*

7.4.4 The LRBF Algorithm

We present our Learning Robustness from Bounded Failure (LRBF) algorithm which uses Confidence Supports $\hat{\mathbb{W}}^j$ from Section 7.4.3 in MPC optimization problem (7.4). This guarantees satisfaction of (7.10) (i.e., design requirement (D1)) by system (2.9) in closed-loop with controller (7.6).

Algorithm 5 Learning Robustness with Bounded Failure (LRBF)

Inputs: $\mathbb{Z}, \hat{\mathbb{W}}^1, x_S$.
for $j = 2, \ldots, J$ **do**
Computing Confidence Support $\hat{\mathbb{W}}^j$
  for $q = 1, \ldots, d$ **do**
    Compute $(1 - \frac{\alpha}{d})$-confidence region $\hat{\Theta}_q^j$ for θ_q
    Compute $\hat{\mathbb{W}}_q^j = \cup_{\bar{\theta}_q \in \hat{\Theta}_q} \text{Supp}(\mathcal{P}_{\bar{\theta}_q}^q)$
  end for
  Set $\hat{\mathbb{W}}^j = \hat{\mathbb{W}}_1^j \times \cdots \times \hat{\mathbb{W}}_d^j$
Solving MPC problem (7.4) using $\hat{\mathbb{W}}^j$
  for $t = 0,1, \ldots, T-1$ **do**
    Apply $v_{t|t}^{j,\star}$ from (7.6) with $\hat{\mathbb{W}}^j$ as uncertainty
  end for
end for

Remark 7.6 *We assume that for all iterations $j \in [J]$, at timestep $t = 0$, MPC problem (7.4) is feasible with disturbance supports $\hat{\mathbb{W}}^j$ constructed in Algorithm 5. This guarantees that we are able to collect at least one data point in each iteration to update Confidence Support $\hat{\mathbb{W}}^j$ while satisfying (7.10). In case such an assumption is not satisfied, $\hat{\mathbb{W}}^j$ can be scaled down (for e.g., by increasing α).*

Remark 7.7 *The convergence of $\hat{\mathbb{W}}^j$ to the true support $\mathbb{W}$ can be sped up by keeping the iteration running until timestep T despite State Constraint Failure. This can be done by introducing slack variables in MPC problem (7.4). Details can be found in the Appendix.*

7.4.5 Case Studies

We now demonstrate our approach for two parametric distribution families: (i) uniform distribution, and (ii) truncated normal distribution.

Uniform Distribution.

Consider the uniform distribution with hyper-rectangle support $\mathbb{W} = [-\theta_1, \theta_1] \times \cdots \times [-\theta_d, \theta_d]$. Then we have,

$$\mathcal{P}^q_{\theta_q} = \text{Unif}(-\theta_q, \theta_q),\ q \in [d].$$

Let $\bar{w}^j(q) = \max_{\bar{w} \in w^{1:j-1}} |\bar{w}|,\ q \in [d]$ and let $\mathcal{T}^j = \sum_{i=1}^{j-1} T^i$. The following set turns out to be a $(1 - \frac{\alpha}{d})$-confidence interval for θ_q,

$$\hat{\Theta}^j_q = \left[\bar{w}^j(q), \frac{\bar{w}^j(q)}{\left(\frac{\alpha}{d}\right)^{1/\mathcal{T}^j}} \right].$$

A derivation of the above confidence interval can be found in the Appendix. Using Lemma 7.1, we have the $(1-\alpha)$-Confidence Support $\hat{\mathbb{W}}^j = \hat{\mathbb{W}}^j_1 \times \cdots \times \hat{\mathbb{W}}^j_d$, where

$$\hat{\mathbb{W}}^j_q = \left[-\frac{\bar{w}^j(q)}{\left(\frac{\alpha}{d}\right)^{1/\mathcal{T}^j}}, \frac{\bar{w}^j(q)}{\left(\frac{\alpha}{d}\right)^{1/\mathcal{T}^j}} \right]. \tag{7.22}$$

Remark 7.8 *This can be extended to the asymmetric case with $\mathcal{P}^q_{\theta_q} = \text{Unif}(-\theta^1_q, \theta^2_q)$. In this case, there is no analytical expression for the Confidence Support but it can be computed numerically.*

Truncated Normal Distribution.

Consider the truncated normal distribution with mean μ_q, variance σ_q^2, and support $[\mu_q - 3\sigma_q, \mu_q + 3\sigma_q]$, i.e.,

$$\mathcal{P}^q_{\theta_q} = \mathcal{N}_{\text{trunc}}(\mu_q, \sigma_q^2, 3),\ q \in [d].$$

As the distribution is fully specified by μ_q and σ_q, we have $\theta_q = [\mu_q, \sigma_q]^\top$. Although it is difficult to derive exact confidence intervals in this case, approximate confidence intervals for μ_q and σ_q can be computed via the Bootstrap [109, Chapter 13]. Let $[\mu^j_{\min}(q), \mu^j_{\max}(q)]$ and

$[\sigma^j_{\min}(q), \sigma^j_{\max}(q)]$ denote the $(1-\frac{\alpha}{2d})$-Bootstrap confidence intervals for μ_q and σ_q respectively. By union bound, we have the following approximate $(1-\frac{\alpha}{d})$-confidence interval for θ_q,

$$\begin{aligned}\hat{\Theta}^j_q = \{[\mu, \sigma]^\top |\ \mu \in [\mu^j_{\min}(q), \mu^j_{\max}(q)],\\ \sigma \in [\sigma^j_{\min}(q), \sigma^j_{\max}(q)]\},\end{aligned}$$

which gives us an approximate $(1-\alpha)$-Confidence Support $\hat{\mathbb{W}}^j = \hat{\mathbb{W}}^j_1 \times \cdots \times \hat{\mathbb{W}}^j_d$, where

$$\hat{\mathbb{W}}^j_q = [\mu^j_{\min}(q) - 3\sigma^j_{\max}(q), \mu^j_{\max}(q) + 3\sigma^j_{\max}(q)]. \tag{7.23}$$

7.5 Numerical Simulations

In this section we find approximate solutions to the following iterative optimal control problem in receding horizon:

$$\begin{aligned}&V^{j,\star}(x_S) = \\ &\min_{u^j_0, u^j_1(\cdot), \ldots} \sum_{t=0}^{T-1} 10 \left\|\bar{x}^j_t - x_{\text{ref}}\right\|_2^2 + 2\left\|u^j_t(\bar{x}^j_t)\right\|_2^2\\ &\text{s.t.,}\\ &\qquad x^j_{t+1} = Ax^j_t + Bu^j_t(x^j_t) + w^j_t,\\ &\qquad \bar{x}^j_{t+1} = A\bar{x}^j_t + Bu^j_t(\bar{x}^j_t),\\ &\qquad \begin{bmatrix}-30\\-30\\-40\end{bmatrix} \le \begin{bmatrix}x^j_t\\ u^j_t(x^j_t)\end{bmatrix} \le \begin{bmatrix}30\\30\\40\end{bmatrix}, \forall w^j_t \in \mathbb{W},\\ &\qquad x^j_0 = x_S,\ t = 0, 1, \ldots, T-1.\end{aligned}$$

We consider two parametric distributions:

$$\mathcal{P}^q_{\theta_q} = \text{Unif}(-3, 3), \tag{7.24a}$$

$$\mathcal{P}^q_{\theta_q} = \mathcal{N}_{\text{trunc}}(0, 1, 3), \tag{7.24b}$$

with $q \in \{1, 2\}$. In both cases, $\mathbb{W} = [-3, 3] \times [-3, 3]$. We construct Bootstrap confidence intervals for the truncated normal case by re-sampling 1000 times. System matrices $A = \begin{bmatrix}1.2 & 1.3\\ 0 & 1.5\end{bmatrix}$ and $B = [0, 1]^\top$ are known. We solve the above optimization problem with the initial state $x_S = [0, 0]^\top$ and reference point $x_{\text{ref}} = [27, 27]^\top$ for task duration $T = 20$ steps over $J = 30$ iterations. Algorithm 5 is implemented with a control horizon of $N = 4$, and the feedback gain K in (7.6) is chosen to be the optimal LQR gain for system $x^+ = (A + BK)x$ with parameters $Q_{\text{LQR}} = 10I_2$ and $R_{\text{LQR}} = 2$. The source code is available at https://github.com /monimoyb/LRBF. The goal is to show:

- Design specification (D1) is satisfied. Consequently, a lower probability of Disturbance Support Failure across all iterations using support $\hat{\mathbb{W}}^j$ from Algorithm 5, compared to that from the convex hull support estimate $C^{\text{hull}}(w^{1:j-1})$.
- The performance loss $[\text{PL}]^j$ rapidly approaches 0 within the first few iterations. However, in the initial iterations, there is a significant trade-off between a desired upper bound α on probability of State Constraint Failure and average closed-loop cost $\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^{1:j})]$ (defined in (7.8)). That is, lower the upper bound α, higher is the average closed-loop cost in the initial iterations. This suggests the need for tailoring the confidence level $(1-\alpha)$ in Algorithm 5 according to the application at hand.

7.5.1 Bounding the Probability of Failure (D1)

In this section, we demonstrate satisfaction of design specification (D1) by Algorithm 5 and compare the probability of Disturbance Support Failure $\mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j)$ for any timestep t in the j^{th} iteration, with $\hat{\mathbb{W}}^j$ obtained using Algorithm 5 and $\hat{\mathbb{W}}^j = C^{\text{hull}}(w^{1:j-1})$. This probability is estimated by averaging over 100 Monte Carlo draws of disturbance samples $w^{1:J}$, i.e.,

$$\mathbb{P}(w_t^j \notin \mathbb{W}^j) \approx \frac{1}{100} \sum_{\tilde{m}=1}^{100} (\mathbf{1}_{\mathcal{F}}(w_t^j))^{\star\tilde{m}},$$

where

$$(\mathbf{1}_{\mathcal{F}}(w_t^j))^{\star\tilde{m}} = \begin{cases} 1, \text{ if } w_t^j \notin (\hat{\mathbb{W}}^j)^{\star\tilde{m}} | (w^{1:j-1})^{\star\tilde{m}}, \\ 0, \text{ otherwise,} \end{cases}$$

and $(\cdot)^{\star\tilde{m}}$ represents the $\tilde{m}^{\text{th}}$ Monte Carlo sample. Fig. 7.1 shows this comparison for uniformly distributed disturbance (7.24a). Using LRBF to construct Confidence Supports $\hat{\mathbb{W}}^j$ allows for lowering $\mathbb{P}(w_t^j \notin \hat{\mathbb{W}}^j)$, i.e., probability of $[\text{DSF}]_t^j$ as defined in (7.11) below a user specified bound α, as opposed to simply utilizing $\hat{\mathbb{W}}^j = C^{\text{hull}}(w^{1:j-1})$. We plot the probability of $[\text{DSF}]_t^j$ for 2 different values of $\alpha = 0.05$ and $\alpha = 0.70$. We see that for $\alpha = 0.05$ the probability of $[\text{DSF}]_t^j$ with LRBF is on average 94% smaller than that from the convex hull support estimate for all iterations $j \in [30]$. Similarly for $\alpha = 0.70$, the probability of $[\text{DSF}]_t^j$ is on average 61% lower than that with the convex hull support estimate across all $j \in [30]$.

The same trend is seen in Fig. 7.2 for truncated normal distribution (7.24b), where probability of $[\text{DSF}]_t^j$ is at least 99% and 96% lower than convex hull support estimate for $\alpha = 0.05$ and $\alpha = 0.70$ respectively until iteration $j = 3$, and reaches a value of 0 for both values of α afterwards. The above trend in probability of $[\text{DSF}]_t^j$ is explained by Proposition 7.1, which relates the desired confidence $(1-\alpha)$ for support $\hat{\mathbb{W}}^j$ to the probability of $[\text{DSF}]_t^j$. Moreover, from Fig. 7.1 and Fig. 7.2 we see that in practice probability of $[\text{DSF}]_t^j$ is always at least $60\% - 80\%$ lower than corresponding chosen α. This highlights satisfaction of (D1) and also the conservatism in Proposition 7.1 arising from the upper bound in (7.20).

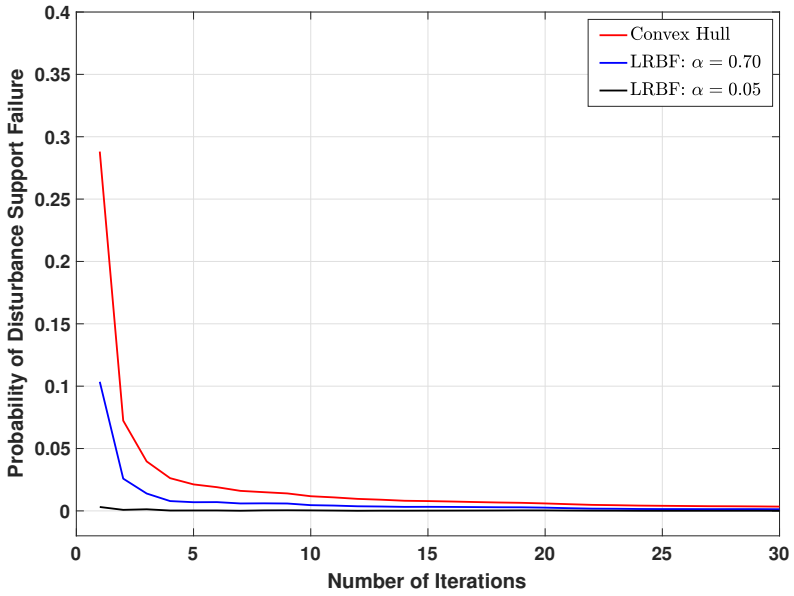


Figure 7.1: Probability of Disturbance Support Failure vs iteration number for uniformly distributed disturbance on $\mathbb{W}$.

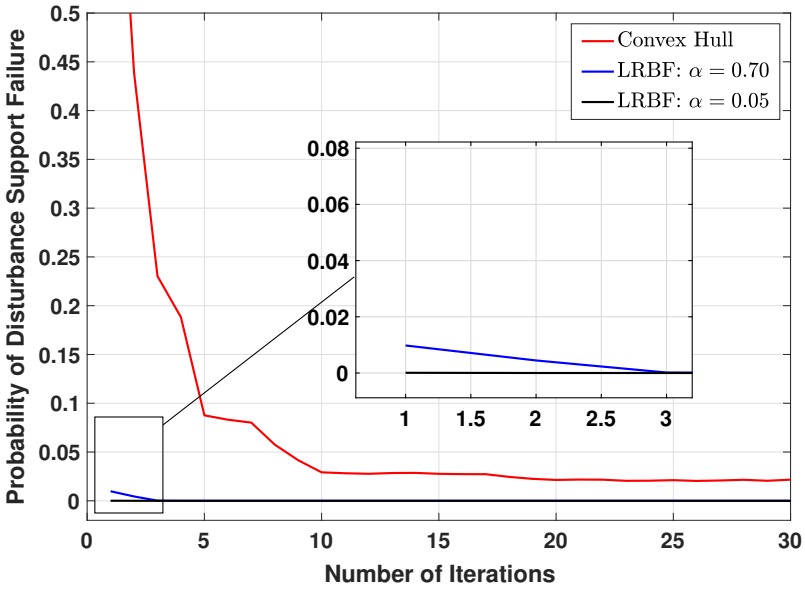


Figure 7.2: Probability of Disturbance Support Failure vs iteration number for truncated normal distribution of disturbance on $\mathbb{W}$.

7.5.2 Performance Loss Reduction Over Iterations

In Fig. 7.3 and Fig. 7.4, we approximate the average closed-loop cost $\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^{1:j})]$ of the j^{th} iteration by taking an empirical average over 100 Monte Carlo draws of $w^{1:J}$ as,

$$\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^{1:j})] \approx \frac{1}{100}\sum_{\tilde{m}=1}^{100} \hat{\mathcal{V}}^j(x_S, (w^{1:j})^{\star\tilde{m}}), \tag{7.25}$$

for $\alpha = 0.05$, and $\alpha = 0.70$. The cost values are normalized by $\mathcal{V}^\star(x_S)$, which denotes the empirical average closed-loop cost of the j^{th} iteration if $\mathbb{W}$ had been known, i.e., $\hat{\mathbb{W}}^j = \mathbb{W}$. For both cases of α, we see that in Fig. 7.3 and Fig. 7.4 the average closed-loop cost rapidly approaches $\mathcal{V}^\star(x_S)$. For (7.24a) in Fig. 7.3, cost (7.25) approaches to within 0.5% of $\mathcal{V}^\star(x_S)$ after just 5 iterations whereas for (7.24b) in Fig. 7.4, it is within 3% of $\mathcal{V}^\star(x_S)$ in the same duration.

However, the average closed-loop cost incurred in earlier iterations has a trade-off with desired α. This trade-off is also highlighted in Fig. 7.3 and Fig. 7.4 for (7.24a) and (7.24b) respectively. We see from Fig. 7.3 and Fig. 7.4 that for lower value of probability of $[\text{SCF}]_t^j$ with $\alpha = 0.05$, we pay a maximum of 13% higher average closed-loop cost for (7.24a), and a maximum of 10% higher average closed-loop cost for (7.24b) compared to $\mathcal{V}^\star(x_S)$ until iteration $j = 5$. Allowing for higher probability of $[\text{SCF}]_t^j$ with $\alpha = 0.70$ proves to be cost-efficient, where we only pay a maximum of 0.3% higher average closed-loop cost for (7.24a), and a maximum of 4% higher average closed-loop cost for (7.24b) compared to $\mathcal{V}^\star(x_S)$ in the same duration. This essentially reflects the key trade-off between specifications (D1) and (D2) in the initial iterations. Thus, the upper bound α of $[\text{SCF}]_t^j$ must be chosen in an application-specific manner.

7.6 Chapter Summary

We proposed an approach to design an MPC for constrained linear time-invariant systems performing an iterative task, where the system is subject to a bounded additive disturbance whose distribution support is not exactly known. The goal was to learn to satisfy state and input constraints robustly by constructing and then successively refining estimates of the disturbance distribution support. Using disturbance measurements after each iteration, we constructed *Confidence Support* sets, which contain the true support of the disturbance distribution with a given (high) probability. As more data is collected, the Confidence Supports converge to the true support of the disturbance. This enabled design of an MPC controller that avoids conservative estimate of the disturbance support, while simultaneously bounding the probability of constraint violation. We demonstrated the efficacy of the proposed approach with a detailed numerical example with both uniform and truncated Gaussian distribution of the additive disturbance.

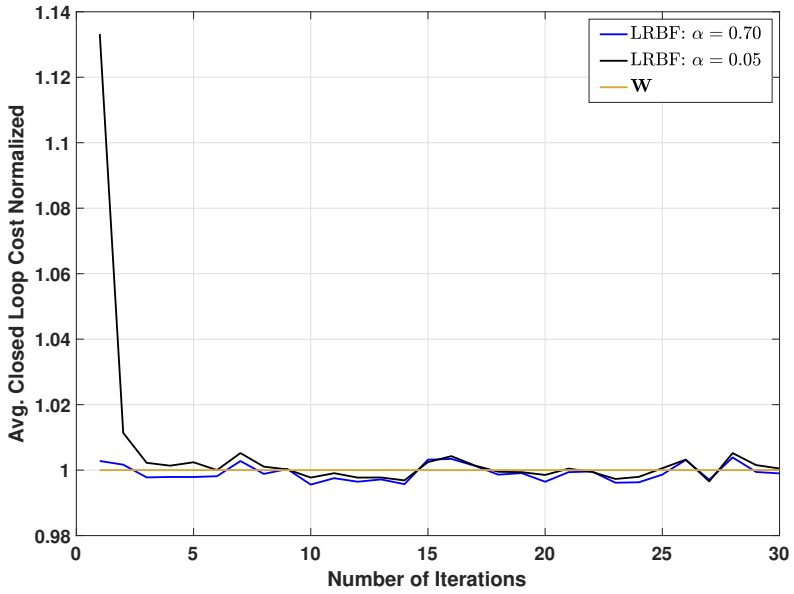


Figure 7.3: Normalized average closed-loop cost (7.25): Uniform disturbance.

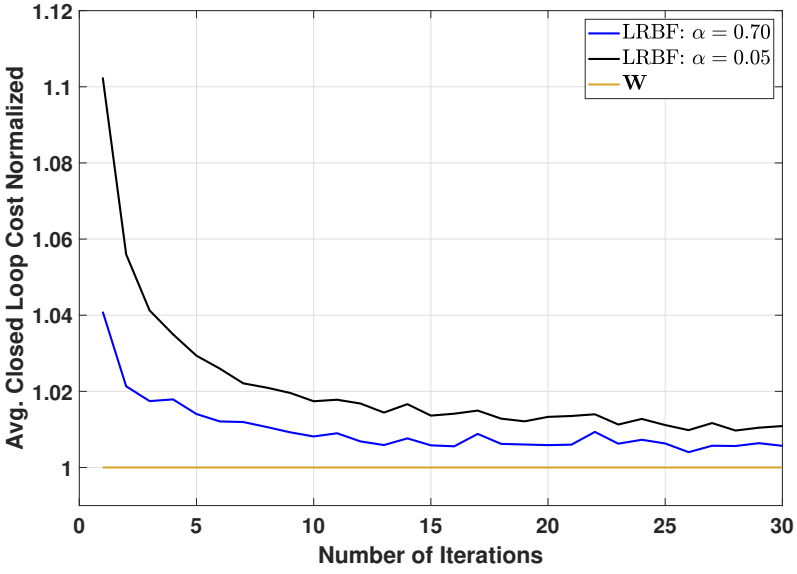


Figure 7.4: Normalized average closed-loop cost (7.25): Truncated normal disturbance.

Appendix

Speeding up Convergence of $\hat{\mathbb{W}}^j$

In order to speed up convergence of $\hat{\mathbb{W}}^j$ in Algorithm 5 to the true support $\mathbb{W}$, the following MPC optimization problem with slack variables is solved:

$$
\begin{aligned}
&\tilde{V}^{\mathrm{MPC},j}_{t\to t+N}(x^j_t, \hat{\mathbb{W}}^j, \hat{\mathcal{X}}^j_N) := \\
&\min_{U^j_t(\cdot)} \quad \sum_{k=t}^{t+N-1} \ell(\bar{x}^j_{k|t}, v^j_{k|t}) + Q(\bar{x}^j_{t+N|t}) + \Lambda\|\mathbf{s}^j_t\|_2^2 \\
&\text{s.t.,} \quad x^j_{k+1|t} = Ax^j_{k|t} + Bu^j_{k|t} + w^j_{k|t}, \\
&\qquad \bar{x}^j_{k+1|t} = A\bar{x}^j_{k|t} + Bv^j_{k|t}, \\
&\qquad u^j_{k|t} = \sum_{l=t}^{k-1} M^j_{k,l|t} w^j_{l|t} + v^j_{k|t}, \\
&\qquad H_x x^j_{k|t} \le h_x + s^j_t,\ H_u u^j_{k|t} \le h_u, \\
&\qquad \hat{Y}^j x^j_{t+N|t} \le \hat{z}^j + \hat{s}^j_t,\ \text{with } \hat{\mathcal{X}}^j_N = \{x : \hat{Y}^j x \le \hat{z}^j\}, \\
&\qquad \mathbf{s}^j_t = [(s^j_t)^\top, (\hat{s}^j_t)^\top]^\top \ge 0, \\
&\qquad \forall w^j_{k|t} \in \hat{\mathbb{W}}^j, \\
&\qquad \forall k = \{t, \dots, t+N-1\}, \\
&\qquad x^j_{t|t} = \bar{x}^j_{t|t} = x^j_t, \Lambda \gg 0,
\end{aligned}
\tag{7.26}
$$

with $\mathbf{s}^j_0 = 0$ (from Remark 7.6), and then closed-loop control law $u^j_t = v^{j,\star}_{t|t}$ is applied to system (2.9). By solving the relaxed optimization problem (7.26) which is feasible for all timesteps $0 \le t \le T-1$ in the j^{th} iteration, we ensure that after each iteration, a set of T additional samples are obtained for the update $\hat{\mathbb{W}}^j \overset{\text{update}}{\longrightarrow} \hat{\mathbb{W}}^{j+1}$. From Section 7.4.3 we can infer that this speeds up the convergence of $\hat{\mathbb{W}}^j$.

Derivation of Confidence Support (7.22)

Consider $w^j_t(q) \overset{\text{iid}}{\sim} \text{Unif}(-\theta_q, \theta_q)$. This implies that $\frac{|w^j_t(q)|}{\theta_q} \overset{\text{iid}}{\sim} \text{Unif}(0,1)$. Let $\bar{w}^j(q) = \max_{\bar{w}\in w^{1:j-1}} |\bar{w}|$. Then, for any $c \in [0,1]$ we have,

$$
\begin{aligned}
\mathbb{P}\left(\frac{\bar{w}^j(q)}{\theta_q} \le c\right) &= \mathbb{P}\left(\bigcap_{\bar{w}\in w^{1:j-1}} \frac{|\bar{w}|}{\theta_q} \le c\right) = \Pi_{\bar{w}\in w^{1:j-1}} \mathbb{P}\left(\frac{|\bar{w}|}{\theta_q} \le c\right), \\
&= c^{\mathcal{T}^j},
\end{aligned}
\tag{7.27}
$$

where (7.27) follows as $\bar{w} \in w^{1:j-1}$ are independent. Setting $c = \alpha_q^{\frac{1}{T^j}}$, we have

$$\mathbb{P}\left(\frac{\bar{w}^j(q)}{\theta_q} \leq \alpha_q^{\frac{1}{T^j}}\right) = \alpha_q, \text{ and therefore } \mathbb{P}\left(\alpha_q^{\frac{1}{T^j}} \leq \frac{\bar{w}^j(q)}{\theta_q} \leq 1\right) = 1 - \alpha_q,$$

which gives us

$$\mathbb{P}\left(\bar{w}^j(q) \leq \theta_q \leq \frac{\bar{w}^j(q)}{\alpha_q^{\frac{1}{T^j}}}\right) = 1 - \alpha_q.$$

Setting $\alpha_q = \frac{\alpha}{d}$ and using Lemma 7.1 completes the derivation.

Chapter 8

Learning Environment Constraints in Robust MPC

This chapter is based on the published work [41]. In this chapter, we propose an algorithm to design a safe controller for an uncertain system while learning polyhedral state constraints. We consider a linear time-invariant system of the form (M1) in Chapter 2, performing an iterative task. The environment constraints of the task are assumed polyhedral, characterized by a set of hyperplanes, some of which are unknown to the control designer. We assume that violations of the unknown constraints can be directly measured from closed-loop trajectories.

8.1 Summary of Contributions

Our algorithm iteratively constructs estimates of the unknown constraints using collected system trajectories. These estimates are then used to design a robust MPC controller [55, 31] for safely achieving the control task despite the uncertainty. The main contributions of this chapter are as follows:

- Given a user-specified upper bound ϵ on the probability of violating the true constraint set $\mathbb{Z}$ within any j^{th} task iteration, we construct constraint estimates $\hat{\mathbb{Z}}^j$ from previously collected closed-loop task data, using convex hull operations (for $\epsilon = 0$) or a Support Vector Machine (SVM) classifier (for $\epsilon \in (0, 1)$). We then design an MPC controller to robustly satisfy $\hat{\mathbb{Z}}^j$ along the j^{th} iteration, for all possible additive disturbance values.
- When $\hat{\mathbb{Z}}^j$ is formed with the SVM classification approach (for $\epsilon \in (0, 1)$), we provide an explicit number of successful task iterations to obtain before the estimated set $\hat{\mathbb{Z}}^j$ is deemed safe with respect to ϵ. Here, "successful task iterations" refers to closed-loop trajectories satisfying the unknown constraints $\mathbb{Z}$.
- When $\hat{\mathbb{Z}}^j$ is formed using the convex hull approach (for $\epsilon = 0$), we show how to design a robust MPC that provides satisfaction of the true constraints $\mathbb{Z}$ at all future iterations $k \geq j$.

8.2 Problem Setup

We consider linear time-invariant systems of the form (2.9). We define $H_x \in \mathbb{R}^{s\times n}$, $h_x \in \mathbb{R}^s$, $H_u \in \mathbb{R}^{o\times m}$, and $h_u \in \mathbb{R}^o$, and formulate the state and input constraints imposed by the task environment for all timesteps $t \geq 0$ as (7.2). Our goal is to design a controller that, at each iteration j, aims to solve the finite horizon robust optimal control problem (7.3). In this work we consider constraints of the form:

$$H_x = \begin{bmatrix} H_x^{\mathrm{b}} \\ H_x^{\mathrm{ub}} \end{bmatrix}, h_x = \begin{bmatrix} h_x^{\mathrm{b}} \\ h_x^{\mathrm{ub}} \end{bmatrix},$$

where the superscripts $\{\mathrm{b}, \mathrm{ub}\}$ denote the known and unknown parts of the constraints, respectively. That is to say, we consider a scenario in which we only know a subset of the system's environment constraint set. At the beginning of the j^{th} task iteration we construct approximations of H_x and h_x, denoted as $\hat{H}_x^j$ and $\hat{h}_x^j$, respectively, using closed-loop trajectories of the system from previous task iterations. The estimated constraints form a safe set estimate $\hat{\mathbb{Z}}^j$:

$$\hat{\mathbb{Z}}^j := \{(x,u) : \hat{H}_x^j x \leq \hat{h}_x^j, H_u u \leq h_u\}. \tag{8.1}$$

These estimates are refined iteratively using new data as the system continues to perform the task, and are used to solve an estimate of (7.3).

8.3 Iterative MPC Problem

Since the true constraint set $\mathbb{Z}$ is not completely known, we use our estimate $\hat{\mathbb{Z}}^j$ built from data and formulate this MPC problem as:

$$\begin{aligned}
&V_{t\to t+N}^{\mathrm{MPC},j}(x_t^j, \hat{\mathbb{Z}}^j, \hat{\mathcal{X}}_N^j) := \\
&\quad \min_{U_t^j(\cdot)} \quad \sum_{k=t}^{t+N-1} \ell(\bar{x}_{k|t}^j, v_{k|t}^j) + Q(\bar{x}_{t+N|t}^j) \\
&\quad \text{s.t.,} \quad x_{k+1|t}^j = Ax_{k|t}^j + Bu_{k|t}^j + w_{k|t}^j, \\
&\qquad\quad \bar{x}_{k+1|t}^j = A\bar{x}_{k|t}^j + Bv_{k|t}^j, \\
&\qquad\quad u_{k|t}^j = \sum_{l=t}^{k-1} M_{k,l|t}^j w_{l|t}^j + v_{k|t}^j, \\
&\qquad\quad \hat{H}_x^j x_{k|t}^j \leq \hat{h}_x^j,\ H_u u_{k|t}^j \leq h_u, \\
&\qquad\quad x_{t|t}^j = \bar{x}_{t|t}^j, \\
&\qquad\quad x_{t+N|t}^j \in \hat{\mathcal{X}}_N^j, \\
&\qquad\quad \forall w_{k|t}^j \in \mathbb{W}, \forall k = \{t, \ldots, t+N-1\},
\end{aligned} \tag{8.2}$$

where $U_t^j(\cdot) = \{u_{t|t}^j, \ldots, u_{t+N-1|t}^j(x_{t+N-1|t})\}$, x_t^j is the measured state at timestep t, $x_{k|t}^j$ is the predicted state at timestep k, obtained by applying predicted policies $\{u_{t|t}^j, \ldots, u_{k-1|t}^j(x_{k-1|t})\}$ to system (2.9). We denote the disturbance-free nominal state and corresponding input as $\{\bar{x}_{k|t}^j, v_{k|t}^j\}$ with $v_{k|t}^j = u_{k|t}^j(\bar{x}_{k|t}^j)$. In (8.2) we have used the affine disturbance feedback parametrization to tackle challenge (C2) introduced in Chapter 2. The MPC controller minimizes the cost over the predicted nominal trajectory $\left\{\{\bar{x}_{k|t}^j, v_{k|t}^j\}_{k=t}^{t+N-1}, \bar{x}_{t+N|t}^j\right\}$, which is comprised of a positive definite stage cost $\ell(\cdot,\cdot)$ and terminal cost $Q(\cdot)$. We use state feedback $u_t^j = Kx_t^j$ with $(A+BK)$ being stable to construct a terminal set $\hat{\mathcal{X}}_N^j = \{x \in \mathbb{R}^n : \hat{Y}^j x \leq \hat{z}^j,\ \hat{Y}^j \in \mathbb{R}^{r^j \times n},\ \hat{z}^j \in \mathbb{R}^{r^j}\}$, which is the $(T-N)$ step robust controllable set to the set of state constraints in (8.1) under the terminal policy, with the properties:

$$\begin{aligned}
&\hat{\mathcal{X}}_N^j \subseteq \{x \mid (x, Kx) \in \hat{\mathbb{Z}}^j\},\\
&\hat{H}_x^j((A+BK)^i x + \sum_{\tilde{i}=0}^{i-1}(A+BK)^{i-\tilde{i}-1} w_{\tilde{i}}) \leq \hat{h}_x^j,\\
&H_u(K((A+BK)^i x + \sum_{\tilde{i}=0}^{i-1}(A+BK)^{i-\tilde{i}-1} w_{\tilde{i}})) \leq h_u,\\
&\forall x \in \hat{\mathcal{X}}_N^j,\ \forall w_i \in \mathbb{W},\ \forall i = 1, 2, \ldots, (T-N).
\end{aligned} \tag{8.3}$$

After solving (8.2) at timestep t of the j^{th} iteration, we apply

$$u_t^j = v_{t|t}^{j,\star} \tag{8.4}$$

to system (2.9). We then resolve the problem (8.2) again at the next $(t+1)$-th timestep, yielding a receding horizon strategy.

Assumption 8.1 (Well-Posedness of Task) *We assume that given an initial task state x_S, the optimization problem (8.2) is feasible at all timesteps $0 \leq t \leq T-1$ for the true constraint set $\hat{\mathbb{Z}}^j = \mathbb{Z}$ as defined in (7.2), for all iterations $j \in \{1, 2, \ldots\}$. We further assume that $\mathbf{0}_{n\times 1} \in \mathbb{Z}$.*

8.3.1 Successful Task Iterations

At each iteration, the true constraint set $\mathbb{Z}$ is unknown and being estimated with $\hat{\mathbb{Z}}^j$ built from data. Depending on how $\hat{\mathbb{Z}}^j$ is constructed, robust satisfaction of the true constraints (7.2) during an iteration may not be guaranteed. It is thus possible that (7.2) becomes infeasible at some point while solving (8.2) during $0 \leq t \leq T-1$ along any j^{th} iteration. We formalize this with the following definition:

Definition 8.1 (Successful Iteration) *A Successful j^{th} Iteration is defined as the event*

$$[\text{SI}]^j : H_x x_t^j \leq h_x,\ \forall t \in [0, T]. \tag{8.5}$$

That is, an iteration is successful if there are no state constraint violations during $0 \leq t \leq T$. *Otherwise, the iteration is deemed failed; that is, an Iteration Failure event is implicitly defined as* $[\text{IF}]^j = ([\text{SI}]^j)^{\text{c}}$, *where* $([\cdot])^{\text{c}}$ *denotes the complement of an event.*

The probability of a Successful Iteration $[\text{SI}]^j$ is a function of the sets $\hat{\mathbb{Z}}^j$ since x_t^j is the closed-loop trajectory obtained when applying the feedback controller (8.2)-(8.4).

8.3.2 Control Design Objectives

Our aim is not only to keep the probability of $[\text{IF}]^j$ low along each iteration, but also to maintain satisfactory controller performance in terms of cost during successful iterations. Let the closed-loop cost of a successful iteration j under observed disturbance samples w^j be denoted by

$$\hat{\mathcal{V}}^j(x_S, w^j) = \sum_{t=0}^{T-1} \ell(x_t^j, v_{t|t}^{j,\star}),$$

where notation w^j denotes $[w_0^j, w_1^j, \dots, w_{T-1}^j]$. We use the average closed-loop cost $\mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^j)]$ to quantify controller performance. Specifically, our goal is to lower the iteration *performance loss*, defined as

$$[\text{PL}]^j = \mathbb{E}[\hat{\mathcal{V}}^j(x_S, w^j)] - \mathbb{E}[\mathcal{V}^\star(x_S, w^j)], \tag{8.6}$$

where $\mathbb{E}[\mathcal{V}^\star(x_S, w^j)]$ denotes the average closed-loop cost of an iteration if $\mathbb{Z}$ had been known, i.e. if $\hat{\mathbb{Z}}^j = \mathbb{Z}$ for all $j \in \{1, 2, \dots\}$.

To formalize this joint focus on obtaining a low probability of Iteration Failures while maintaining a satisfactory controller performance, we summarize our control design objectives as:

(C1) Design a closed-loop MPC control law (8.4) which ensures that the system (2.9) maintains a user-specified upper bound on the probability of Iteration Failure $[\text{IF}]^j$ (8.5), *for all* iterations $j \in \{1, 2, \dots\}$.

(C2) Minimize $[\text{PL}]^j$ (as defined in (8.6)) at *each* iteration $j \in \{1, 2, \dots\}$, while satisfying objective (C1).

However, as we start the control task from scratch without assuming the initial availability of a large number of trajectory data samples, and it is difficult in general to obtain statistical properties of estimated constraint sets $\hat{\mathbb{Z}}^j$, methods such as [37, 34, 36] cannot be used to satisfy (C1)-(C2) directly. We therefore relax the above two specifications and formulate two control design specifications (D1) and (D2) in the next section.

8.4 Iterative Constraint Learning

We consider the following design specifications:

(D1) Design a closed-loop MPC control law (8.4) which ensures that the system (2.9) maintains a user-specified upper bound ϵ on the probability of Iteration Failure, *after* some iteration $j \in \{1, 2, \dots\}$.

(D2) Minimize $[\text{PL}]^j$ (as defined in (8.6)) *after* some iteration $j \in \{1, 2, \dots\}$.

We wish to find the smallest index $\bar{j}$, such that (D1) and (D2) are satisfied for all $j \geq \bar{j}$. The design specifications (D1)-(D2) indicate that the approach to construct estimated state constraint sets proposed, is our best possible attempt to satisfy (C1)-(C2), given the information available at each iteration j.

Assumption 8.2 (Feasibility Classification) *Given a system state trajectory, we assume that a classifier is available to check the feasibility of each point in the trajectory based on whether it satisfies the true state constraints in (7.2). This classifier returns a corresponding sequence of feasibility flags.*

Assumption 8.3 (Simulator) *We assume that each iteration is run until completion at timestep T, and that state constraint satisfaction as described in Assumption 8.2 is checked only at the end of the simulation.*

We note that Assumption 8.3 could be relaxed in several ways. For example, constraint satisfaction could be checked in real-time and the simulations stopped if violations occur. One could also run physical experiments and check the feasibility of (7.2) in real-time, by observing if the physical experiment fails. Some constraint violations may be hard to evaluate during physical experiments, but this discussion goes beyond the scope of this dissertation.

8.4.1 Constructing Constraint Estimates $\hat{\mathbb{Z}}^j$

We show how the estimated constraint sets $\hat{\mathbb{Z}}^j$ are constructed in order to satisfy the design specifications (D1) and (D2). This process depends on the user-specified upper bound ϵ on the probability of Iteration Failure. To satisfy (D1) we search for the smallest $\bar{j}$, such that

$$\mathbb{P}([\text{IF}]^j) \leq \epsilon, \tag{8.7}$$

for all $j \geq \bar{j}$, where $\epsilon \in (0, 1)$ is the bound on the probability of Iteration Failure. At the start of the first iteration, $j = 1$, we use only the known information about the imposed constraints:

$$\hat{\mathbb{Z}}^1 := \{(x, u) : H_x^{\text{b}} x \leq h_x^{\text{b}}, H_u u \leq h_u\}. \tag{8.8}$$

Next, consider any $j \in \{1, 2, \dots\}$. Let the closed-loop realized states collected until the end of the j^{th} iteration be

$$\mathbf{x}^{1:j} = [x_0^{1:j}, x_1^{1:j}, \dots, x_T^{1:j}], \tag{8.9}$$

where $x_i^{1:j} \in \mathbb{R}^{n \times j}$ is a matrix containing all states corresponding to timestep i from the first j iterations. Let $f^j(x) : \mathbb{R}^n \mapsto \mathbb{R}$ denote a curve that separates the points in (8.9) according to whether they satisfy all true state constraints in (7.2), such that $f^j(\mathbf{0}_{n\times 1}) \leq 0$. Based on Assumption 8.2, such a binary classification curve can be obtained with supervised learning techniques. We use a kernelized Support Vector Machine algorithm [42, Chapter 12].

Let a polyhedral inner approximation[1] of the intersection of $f^j(x) \leq 0$ and the known state constraints in $\hat{\mathbb{Z}}^1$ be given by:

$$\begin{aligned} \hat{\mathcal{P}}_{\text{svm}}^{j+1} &= \{x : \hat{H}_{x,\text{svm}}^{j+1} x \leq \hat{h}_{x,\text{svm}}^{j+1}\} \\ &= \{x : f^j(x) \leq 0\} \cap \{x : H_x^{\text{b}} x \leq h_x^{\text{b}}\}. \end{aligned} \tag{8.10}$$

We then use (8.10) to form the constraint set estimates for the following iteration:

$$\hat{\mathbb{Z}}_{\text{svm}}^{j+1} := \{(x, u) : \hat{H}_{x,\text{svm}}^{j+1} x \leq \hat{h}_{x,\text{svm}}^{j+1}, H_u u \leq h_u\}, \tag{8.11}$$

setting $\hat{\mathbb{Z}}^{j+1} = \hat{\mathbb{Z}}_{\text{svm}}^{j+1}$ in our robust optimization problem (8.2) for $j \in \{1, 2, \dots\}$. In other words, at each iteration $j > 1$, the estimated state constraints in $\hat{\mathbb{Z}}^j$ are formed out of the SVM classification boundary learned from all previous state trajectories, intersected with the known state constraints.

Remark 8.1 *In case the set $\hat{\mathbb{Z}}_{\text{svm}}^{j+1}$ in (8.11) yields either infeasibility of (8.2) or an empty terminal set $\hat{\mathcal{X}}_N^{j+1}$ for any iteration $j \in \{1, 2, \dots\}$, the set of estimated state constraints can be scaled appropriately until feasibility of (8.2) is obtained. Such scaling is not further analyzed in the remaining sections of this chapter.*

Since the estimated constraint sets (8.11) are not necessarily inner approximations of the true unknown constraints (7.2), the closed-loop state trajectories in future iterations may result in Iteration Failures with a nonzero probability. In the following proposition we quantify the probability of an Iteration Failure, given a $\hat{\mathbb{Z}}^{\bar{j}}$, for some $\bar{j} \in \{1, 2, \dots\}$.

Proposition 8.1 *Consider $\hat{\mathbb{Z}}_{\text{svm}}^1 = \hat{\mathbb{Z}}^1$ from (8.8) for $\bar{j} = 1$ or a constraint estimate set $\hat{\mathbb{Z}}_{\text{svm}}^{\bar{j}}$ from (8.11) formed using trajectories up to iteration $\bar{j} - 1$ for $\bar{j} > 1$. Let this set $\hat{\mathbb{Z}}_{\text{svm}}^{\bar{j}}$ be used as the constraint estimate set for the next N_{it} task iterations, beginning with iteration $\bar{j}$. If for a chosen $\epsilon \in (0, 1)$ and $0 < \beta \ll 1$, Successful Iterations are obtained for the next $N_{\text{it}} \geq \frac{\ln 1/\beta}{\ln 1/(1-\epsilon)}$ iterations, then $\mathbb{P}([\text{IF}]^j) \leq \epsilon$ with confidence at least $1 - \beta$ for all subsequent task iterations $j \geq \bar{j}$ using $\hat{\mathbb{Z}}^j = \hat{\mathbb{Z}}_{\text{svm}}^{\bar{j}}$.*

[1] Approximation techniques are elaborated in Section 8.5.

Proof *See Appendix.*

Proposition 8.1 requires that the polytope $\hat{\mathcal{P}}^{j+1}_{\text{svm}}$ for $j \in \{1, 2, \dots\}$ is updated *only* if new violation points for constraints (7.2) are seen at the end of an iteration j. This update strategy is highlighted in Algorithm 6. If no violations are seen for N_{it} successive iterations, a probabilistic safety certificate is provided and Algorithm 6 is terminated.

8.4.2 Safety vs Performance Trade-Off

Proposition 8.1 proves that constructing estimated constraint sets as per (8.11), can result in satisfaction of (8.7) for some $\epsilon \in (0, 1)$. However, for certain applications, violations of constraints (7.2) may be too expensive to allow for a nonzero probability of failure, and we instead require $\epsilon = 0$. In such cases, we can utilize the closed-loop system trajectories for obtaining *guaranteed* inner approximations of (7.2), so that $\mathbb{P}([\text{IF}]^j) = 0$ for all future iterations $j \geq \bar{j}$, for some $\bar{j}$ to be determined.

Recalling (8.9), let the closed-loop realized states collected until the end of the j^{th} iteration be denoted as

$$\mathbf{x}^{1:j} = [x_0^{1:j}, x_1^{1:j}, \dots, x_T^{1:j}], \tag{8.12}$$

and let $\hat{\mathbf{x}}^j$ denote the collection of states from (8.12) which satisfy all true state constraints in (7.2). Then an inner approximation the of state constraints in (7.2) is provided by the polyhedron:

$$\begin{aligned} \hat{\mathcal{P}}^{j+1}_{\text{cvx}} &= \{x : \hat{H}^{j+1}_{x,\text{cvx}} x \leq \hat{h}^{j+1}_{x,\text{cvx}}\} \\ &= \text{conv}([\mathbf{0}_{n\times 1}, \hat{\mathbf{x}}^j]), \end{aligned} \tag{8.13}$$

where $\text{conv}(\cdot)$ denotes the convex hull operator. We can now define

$$\hat{\mathbb{Z}}^{j+1}_{\text{cvx}} := \{(x, u) : \hat{H}^{j+1}_{x,\text{cvx}} x \leq \hat{h}^{j+1}_{x,\text{cvx}}, H_u u \leq h_u\}, \tag{8.14}$$

and use $\hat{\mathbb{Z}}^j = \hat{\mathbb{Z}}^j_{\text{cvx}}$ for $j \in \{2, 3, \dots\}$ in (8.2) as a robust alternative to (8.11).

Proposition 8.2 *If $\hat{\mathbb{Z}}^{\bar{j}}_{\text{cvx}}$ (8.14) yields feasibility of (8.2) for some $\bar{j} \in \{2, 3, \dots\}$, then*

$$\hat{\mathbb{Z}}^{\bar{j}}_{\text{cvx}} \subseteq \mathbb{Z},$$

and $\hat{\mathbb{Z}}^j_{\text{cvx}} = \hat{\mathbb{Z}}^{\bar{j}}_{\text{cvx}}$ for all $j \geq \bar{j}$.

Proof *Let the closed-loop realized states collected until the end of the $(\bar{j}-1)^{\text{th}}$ iteration be*

$$\mathbf{x}^{1:\bar{j}-1} = [x_0^{1:\bar{j}-1}, x_1^{1:\bar{j}-1}, \dots, x_T^{1:\bar{j}-1}], \tag{8.15}$$

and let $\hat{\mathbf{x}}^{\bar{j}-1}$ *be the collection of all trajectory points in* (8.15) *that satisfy the state constraints in* (7.2)*. Following* (8.13) *we form* $\hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}} = \{x : \hat{H}^{\bar{j}}_{x,\text{cvx}} x \leq \hat{h}^{\bar{j}}_{x,\text{cvx}}\}$ *as,*

$$\hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}} = \text{conv}([\mathbf{0}_{n\times 1}, \hat{\mathbf{x}}^{\bar{j}-1}]).$$

By the convexity of the true unknown state constraints (7.2) *and Assumption 8.1, we have that* $\hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}} \subseteq \{x : H_x x \leq h_x\}$*. This implies* $\hat{\mathbb{Z}}^{\bar{j}} \subseteq \mathbb{Z}$*.*

Furthermore, since (8.2) *is feasible at timestep* $t = 0$ *in iteration* $\bar{j}$*,* (8.2) *remains feasible with system* (2.9) *in closed-loop with the MPC controller* (8.6) *at all future timesteps* $t \leq (T-1)$*.*[2] *It follows that* $x^{\bar{j}}_t \in \hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}}$ *for all* $0 \leq t \leq T$*, which implies* $\hat{\mathcal{P}}^{\bar{j}+1}_{\text{cvx}} = \hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}}$ *from* (8.13)*. Extending this argument, we can similarly prove* $\hat{\mathcal{P}}^{j}_{\text{cvx}} = \hat{\mathcal{P}}^{\bar{j}}_{\text{cvx}}$ *for all* $j > \bar{j}$*, which implies* $\hat{\mathbb{Z}}^{j}_{\text{cvx}} = \hat{\mathbb{Z}}^{\bar{j}}_{\text{cvx}}$ *for all* $j > \bar{j}$*. This completes the proof.*

Proposition 8.2 implies that if we find a $\bar{j}$ for which (8.14) yields feasibility of (8.2), then the probability of Iteration Failure at iteration j is exactly 0 for all $j \geq \bar{j}$. Moreover, Proposition 8.2 suggests that after the $\bar{j}^{\text{th}}$ iteration, the constraint estimation update (8.14) can be terminated.

The update strategy (8.14) strictly ensures that $\hat{\mathbb{Z}}^{j}_{\text{cvx}} \subseteq \mathbb{Z}$ for all $j \in \{2, 3, \dots\}$, which is not necessarily true for sets obtained using the SVM method (8.11). However, choosing this robust constraint estimation can increase the performance loss (8.6) over successful iterations after $j \geq \bar{j}$. This is the *safety vs. performance trade-off*, which the user can manage with an appropriate choice of ϵ. Given any $\epsilon \in (0, 1)$ or $\epsilon = 0$, the chosen strategy lowers performance loss while satisfying (D1). Thus we satisfy (D2) with (D1).

Remark 8.2 *Following Remark 8.1, if the optimization problem* (8.2) *is infeasible or the terminal set* $\hat{\mathcal{X}}^{j}_{N}$ *constructed in* (8.3) *using the estimate* $\hat{\mathbb{Z}}^{j}_{\text{cvx}}$ *is empty, one can switch to constraint estimates* (8.11) *and collect additional trajectory data, since* $\hat{\mathcal{P}}^{j_1}_{\text{cvx}} \subseteq \hat{\mathcal{P}}^{j_2}_{\text{cvx}}$*, for any* $2 \leq j_1 < j_2$*.*

8.4.3 The RMPC-ICL Algorithm

We present our Robust MPC with Iterative Constraint Learning (RMPC-ICL) algorithm, which uses the estimated constraint sets $\hat{\mathbb{Z}}^{j}$ from Section 8.4.1 or Section 8.4.2 while solving (8.2) in an iterative fashion. The algorithm terminates upon finding the smallest $\bar{j}$ such that (8.7) is satisfied.

8.5 Numerical Simulations

We verify the effectiveness of the proposed Algorithm 6 in a simulation example. The source code is available at https://github.com /monimoyb/ConstraintLearning. We find

[2] This recursive feasibility property is stated without proof. Interested readers can look into the standard detailed proofs in [6, Chapter 12].

Algorithm 6 RMPC-ICL Algorithm

Initialize: $j = 1, l = 0$, $\hat{\mathbb{Z}}^1_{\text{svm}} = \hat{\mathbb{Z}}^1_{\text{cvx}} = \hat{\mathbb{Z}}^1$ from (8.8)
Inputs: $\mathbb{W}$, ϵ, β, N and $x_0^j = x_S$ for all $j \in \{1, 2, \dots\}$
Data: $\tilde{\mathbf{x}}^1 = [x_0^1, x_1^1, \dots, x_T^1]$, $\hat{\mathcal{P}}^2_{\text{cvx}}$ formed with (8.13);

while $j \geq 2$ **do**
  if Points in $\tilde{\mathbf{x}}^{j-1}$ violate (7.2) **then**
    Construct $\hat{\mathbb{Z}}^j_{\text{svm}}$ with (8.11); construct $\hat{\mathcal{X}}^j_N$ with (8.3);
  else
    $\hat{\mathbb{Z}}^j_{\text{svm}} = \hat{\mathbb{Z}}^{j-1}_{\text{svm}}$;
    $l = l + 1$; (if $l \geq \frac{\ln 1/\beta}{\ln 1/(1-\epsilon)}$, **break**; (8.7) is satisfied)
  end if
  if $\mathbb{P}([\text{IF}]^j) = 0$ desired **then**
    Construct $\hat{\mathbb{Z}}^j_{\text{cvx}}$ with (8.14); construct $\hat{\mathcal{X}}^j_N$ with (8.3);
    if Problem (8.2) is feasible with $\hat{\mathbb{Z}}^j_{\text{cvx}}$ **then**
      Use $\hat{\mathbb{Z}}^j = \hat{\mathbb{Z}}^j_{\text{cvx}}$ for solving (8.2);
      break; ($\mathbb{P}([\text{IF}]^j) = 0$ is satisfied)
    else
      Use $\hat{\mathbb{Z}}^j = \hat{\mathbb{Z}}^j_{\text{svm}}$ for solving (8.2);
    end if
  else
    Use $\hat{\mathbb{Z}}^j = \hat{\mathbb{Z}}^j_{\text{svm}}$ for solving (8.2);
  end if
  Set $\tilde{\mathbf{x}}^j = x_S, t = 0$;
  while $0 \leq t \leq T - 1$ **do**
    Solve (8.2) and apply MPC (8.4) to (2.9);
    Collect states and append $\mathbf{x}^j = [\mathbf{x}^j, x^j_{t+1}]$;
    $t = t + 1$
  end while
    $j = j + 1$
end while

approximate solutions to the following iterative optimal control problem in receding horizon:

$$
\begin{aligned}
\min_{u_0^j, u_1^j(\cdot), \dots} \; & \sum_{t=0}^{T-1} 10 \left\| \bar{x}_t^j - x_{\text{ref}} \right\|_2^2 + 2 \left\| u_t^j(\bar{x}_t^j) \right\|_2^2 \\
\text{s.t.}, \quad & \\
& x_{t+1}^j = A x_t^j + B u_t^j(x_t^j) + w_t^j, \\
& \begin{bmatrix} H_x^{\text{b}} \\ H_x^{\text{ub}} \end{bmatrix} x_t^j \leq \begin{bmatrix} 20 \times \mathbf{1}_4 \\ 5 \times \mathbf{1}_2 \end{bmatrix}, \; \forall w_t^j \in \mathbb{W}, \\
& -30 \leq u_t^j(x_t^j) \leq 30, \qquad \forall w_t^j \in \mathbb{W}, \\
& x_0^j = x_S, \; t = 0, 1, \dots, (T-1),
\end{aligned}
\tag{8.16}
$$

where

$$\mathbb{W} = [-0.5, 0.5] \times [-0.5, 0.5],$$
$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \; B = [0, 1]^\top$$

are known. The known and unknown parts of the state constraints are parametrized by the polytopes $\{x : H_x^{\mathrm{b}} x \leq 0\}$ and $\{x : H_x^{\mathrm{ub}} x \leq 0\}$ respectively, where the matrices are given by

$$H_x^{\mathrm{b}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}, \; H_x^{\mathrm{ub}} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

We solve the above optimization problem (8.16) with the initial state $x_S = [-15, 15]^\top$ and reference point $x_{\mathrm{ref}} = [5, 0]^\top$ for task duration $T = 10$ steps over all the iterations. Algorithm 6 is implemented with a control horizon of $N = 4$, and the feedback gain K in (8.3) is chosen to be the optimal LQR gain with parameters $Q_{\mathrm{LQR}} = 10 I_2$ and $R_{\mathrm{LQR}} = 2$. The optimization problems are formulated with YALMIP interface [110] in MATLAB, and we use the Gurobi solver to solve a quadratic program at every timestep for control synthesis. The goal of this section is to show:

- Our approach finds an iteration index $\bar{j}$ such that (8.7) is guaranteed to hold for all iterations $j \geq \bar{j}$.
- Performance loss $[\mathrm{PL}]^j$ over Successful Iterations (after $j \geq \bar{j}$) increases as the tolerable probability ϵ of Iteration Failure is lowered. This highlights the safety vs. performance trade-off.

8.5.1 Bounding the Probability of Iteration Failure

We demonstrate satisfaction of design specification (D1) by Algorithm 6. First, we focus on the SVM-based approach. We choose an SVM classifier with a Radial Basis kernel function [42, Chapter 12]. For introducing exploration properties, the SVM classifier $f^0(x)$ was initially warm-started by exciting the system (2.9) with random inputs and collecting trajectory data for two trajectories. After that the control process was started by solving (8.2). The polytopes $\mathcal{P}_{\mathrm{svm}}^{j+1}$ were generated by taking a convex hull of randomly generated 1000 test points before each iteration, which were classified as $f^j(x_{\mathrm{test}}^j) \leq 0$ for $j \in \{1, 2, \dots\}$.

We consider two cases of tolerable Iteration Failure, with respective probabilities of 30% and 50%, corresponding to $\epsilon = 0.3$ and $\epsilon = 0.5$ (see Table I). The associated estimated constraint sets $\hat{\mathbb{Z}}^{\bar{j}}$ were obtained for $\bar{j} = 5$ and $\bar{j} = 3$ respectively [3]. These sets satisfy design

[3] We note that the exact value of $\bar{j}$, as well as the associated estimated constraint sets, depend on the disturbance sequence. Running this example several times will yield similar results, but not the exact same results.

requirement (D1) and are shown in Fig. 8.1. As expected, the constraint set estimated with $\epsilon = 0.5$ is larger than the set estimated with $\epsilon = 0.3$. Both estimated sets partially violate the true constraint set (outlined in black).

Furthermore, in order to verify the certificate obtained using Proposition 8.1, we run 100 *offline* Monte-Carlo simulations (or *trials*) of iterations by solving (8.2), with $\hat{\mathbb{Z}}^{1:100} = \hat{\mathbb{Z}}^{\bar{j}}$, for each of the above $\hat{\mathbb{Z}}^{\bar{j}}$ sets, and estimate the actual resulting Iteration Failure probability. This probability is estimated by averaging over 100 Monte Carlo draws of disturbance samples $w_{0:T-1} = [w_0, w_1, \ldots, w_{T-1}]$, i.e.,

$$\mathbb{P}(x_{0:T} \notin \hat{\mathbb{Z}}_s^{\bar{j}}) \approx \frac{1}{100} \sum_{\tilde{m}=1}^{100} (\mathbf{1}_{\mathcal{F}}(x_{0:T}))^{\star\tilde{m}},$$

where

$$(\mathbf{1}_{\mathcal{F}}(x_{0:T}))^{\star\tilde{m}} = \begin{cases} 1, & \text{if } x_{0:T} \notin \hat{\mathbb{Z}}_s^{\bar{j}} | (w_{0:T-1})^{\star\tilde{m}}, \\ 0, & \text{otherwise,} \end{cases}$$

and $(\cdot)^{\star\tilde{m}}$ represents the $\tilde{m}^{\text{th}}$ Monte Carlo sample[4]. The values obtained were $\hat{\epsilon} \approx 0.01$ and

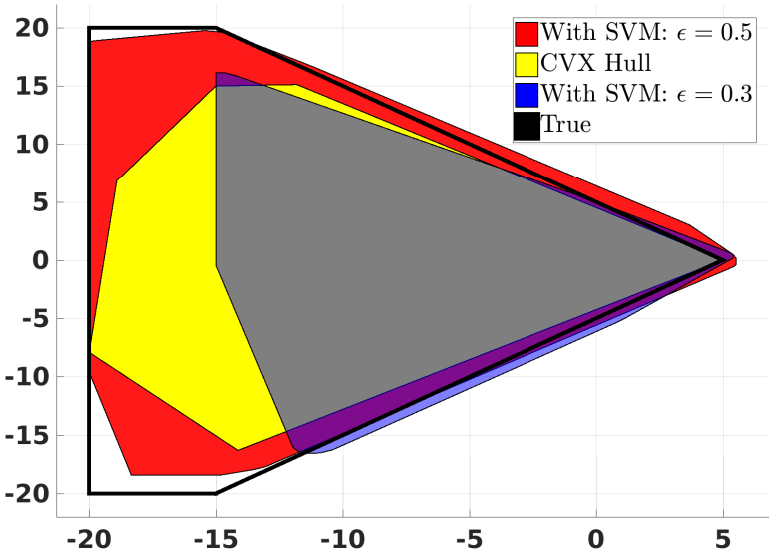


Figure 8.1: Estimated state constraint sets with varying bounds for $\mathbb{P}([\text{IF}]^j)$.

$\hat{\epsilon} \approx 0.04$ for $\epsilon = 0.3$ and $\epsilon = 0.5$ respectively. Thus we see that, in practice, the actual probability of Iteration Failure is about $92\% - 96\%$ lower than the corresponding chosen ϵ. This highlights the conservatism of the bounds given in Proposition 8.1.

We next verify the satisfaction of design requirement (D1) when the estimated constraint sets are obtained using the robust convex hull based approach from Section 8.4.2. We use

[4]For brevity, with slight abuse of notation, we use $\hat{\mathbb{Z}}_s^{\bar{j}}$ to denote the corresponding state constraints.

the same 100 draws of disturbance sequences $w^{1:100} = [w^1_{0:T-1}, w^2_{0:T-1}, \ldots, w^{100}_{0:T-1}]$ as for the SVM-based approach above. The resulting constraint estimate set is shown in Fig. 8.1 and is obtained at $\bar{j} = 4$. Using this set in (8.2) ensures no Iteration Failures for all $j \geq \bar{j}$, as proven in Proposition 8.2. These results from Section 8.5.1 are summarized in Table I.

8.5.2 Safety vs. Performance Trade-Off

For the same Monte Carlo draws of $w^{1:100}$, we approximate the average closed-loop cost $\mathbb{E}[\hat{\mathcal{V}}^{\bar{j}}(x_S, w_{0:T})]$ by taking an empirical average over the 100 Monte Carlo draws,

$$\mathbb{E}[\hat{\mathcal{V}}^{\bar{j}}(x_S, w_{0:T-1})] \approx \frac{1}{100} \sum_{\tilde{m}=1}^{100} \hat{\mathcal{V}}^{\bar{j}}(x_S, (w_{0:T-1})^{\star \tilde{m}}),$$

with $\hat{\mathbb{Z}}^{\bar{j}}$ sets obtained in Fig. 8.1. The cost values are normalized by $\mathcal{V}^\star(x_S)$, which denotes the empirical average closed-loop cost if $\mathbb{Z}$ had been known.

The results are summarized in Table I. We see that the average closed-loop cost shows an inverse relationship with the tolerable Iteration Failure probability ϵ. For lower probabilities of $[\text{IF}]^j$ with $\epsilon = 0.30$, we pay a 3% lower average closed-loop cost compared to $\mathcal{V}^\star(x_S)$. Allowing for higher probability of $[\text{PL}]^j$ with $\epsilon = 0.50$ proves to be cost-efficient, where we pay around 7% lower average closed-loop cost compared to $\mathcal{V}^\star(x_S)$. The cost for the approach in Section 8.4.2 is the highest, with a 4% higher average closed-loop cost compared to $\mathcal{V}^\star(x_S)$. This directly reflects the safety vs. performance trade-off. This also shows that our approach lowers performance loss for any given ϵ, satisfying (D2) with (D1).

Table 8.1: The safety vs performance trade-off.

ϵ	$\bar{j}$	$\hat{\epsilon}$	$\mathbb{E}[\hat{\mathcal{V}}^{\bar{j}}(x_S, w_{0:T-1})]/\mathcal{V}^\star(x_S) \approx$
0	4	0	1.04
0.3	5	0.01	0.97
0.5	3	0.04	0.93

8.6 Chapter Summary

We proposed a framework for an uncertain linear time-invariant system to iteratively learn to satisfy unknown polyhedral state constraints in the environment. From historical trajectory data, we constructed an estimate of the true environment constraints before starting an iteration, which the MPC controller robustly satisfies at all times along the iteration. A *safety* certification is then provided for the estimated constraints, if the true (and unknown) environment constraints are also satisfied by the controller in closed-loop. We further highlight a trade-off between safety and controller performance, numerically demonstrating that a controller designed with estimated constraint sets which are deemed highly safe, result in a higher average closed-loop cost incurred across iterations.

Appendix

Proof of Proposition 8.1

Recall matrices H_x and h_x defined in (7.2). Let us denote $\mathbf{H}_x = H_x \otimes I_T$ and $\mathbf{h}_x = h_x \otimes I_T$. Let $[\mathbf{H}_x]_i$ and $[\mathbf{h}_x]_i$ denote the i^{th} row of $\mathbf{H}_x$ and $\mathbf{h}_x$ respectively. For a fixed initial condition $x_0^{1:j} = x_S$ and a random disturbance realization $\mathbf{w} = [w_0, w_1, \ldots, w_{T-1}]$, consider the corresponding closed-loop trajectory

$$\mathbf{x}(\mathbf{w}) = [x_0^\top, x_1^\top, \ldots, x_T^\top]^\top.$$

Now consider the following function

$$Q(\mathbf{w}) := \max_i \{[\mathbf{H}_x]_i \mathbf{x}(\mathbf{w}) - \mathbf{h}_i\},$$

and then define $\hat{Q}_{N_{\text{it}}} := \max_{j=1,2,\ldots,N_{\text{it}}}\{Q(\mathbf{w}^j)\}$, where $\mathbf{w}^j$ for $j \in \{1, 2, \ldots, N_{\text{it}}\}$ are a collection of independent samples of $\mathbf{w}$ drawn according to $\mathbb{P}$. It follows [111, Theorem 3.1] that, if $N_{\text{it}} \geq \frac{\ln 1/\beta}{\ln 1/(1-\epsilon)}$, then

$$\mathbb{P}^{N_{\text{it}}}\left[\mathbb{P}[Q(\mathbf{w}) > \hat{Q}_{N_{\text{it}}}] \leq \epsilon\right] \geq 1 - \beta.$$

Proposition 8.1 now follows upon setting $\hat{Q}_{N_{\text{it}}} = 0$.

Chapter 9

Playing Cup-and-Ball: An Application of LRBF

This chapter is based on the published work [112]. In this chapter, we demonstrate a fully physics driven model-based hybrid approach for control design in order for a robotic manipulator to learn to play the cup-and-ball game. The controller guarantees a *constrained* motion, while accounting for our best *estimates* of uncertainty in the system model and sensing errors. For obtaining the position of the ball required for closed-loop control design, noisy measurements obtained from a camera are used. The support of this camera noise is refined with data using the tools presented in Chapter 7, which improves the catching capabilities of the robot.

9.1 Summary of Contributions

We use a mixed open-loop and closed-loop control design, motivated by works such as [113, 114, 115]. First, the swing-up phase is designed offline and then an *open-loop* policy is applied to the robotic manipulator. We use a *cart with inverted pendulum model* of the cup-and-ball joint system for swing-up policy design. For this phase, as we solve a constrained finite horizon non-convex optimization problem, we only consider a *nominal* disturbance-free model of the system. The swing-up trajectory is thus designed to ensure that the predicted difference in positions of the ball and the cup vanishes at a future time once the nominal terminal swing-up state is reached and the cup is held fixed.

After a swing-up, we switch to online closed-loop control synthesis once the ball starts its free-fall. We consider presence of only a camera that takes noisy measurements of the ball's position at every timestep. We design the feedback controller in the manipulator's end-effector [116] space. This results in a linear time-invariant (LTI) model for the evolution of the difference between the cup and the ball's positions, thus allowing us to solve convex optimization problems online for control synthesis. In order to guarantee a catch by minimizing the position difference, it is also crucial to ensure that during the free-fall of the ball,

the control actions to the manipulator do not yield a configuration where the string is taut, despite uncertainty in the model and noise in camera position measurements. Uncertainty in the LTI model primarily arises from low level controller mismatches in the manipulator hardware, and an upper bound of this uncertainty is assumed *known*. Bounds on the measurement noise induced by the camera are assumed unknown. We present a method to increase the *probability* of a catch, as the estimate of the support of camera measurement noise distribution is updated. Our contributions are summarized as:

- *Offline*, before the feedback control of the manipulator, we design a swing-up trajectory for the nominal cup-and-ball system that plans the motion of the ball to a state from which a catch control is initiated.
- Using the notion of *Confidence Support* from Chapter 7, which is guaranteed to contain the true support of the camera measurement noise with a specified probability, we use *online* robust feedback control for enforcing bounds on the probability of failed catches.
- With high-fidelity Mujoco simulations and preliminary physical experiments we demonstrate that the manipulator gets better at catching the ball as the support of the camera measurement noise is learned and as the Confidence Support and closed-loop policy are updated.

9.2 Generating A Swing-up Trajectory

The swing-up phase begins with the arm in the home position such that the ball is hanging down at an angle of 0 radians from the vertical plumb line, as seen in Fig. 9.1.

9.2.1 System Modeling

We model the system such that the cup is a planar cart with point-mass m_c and the ball acts as a rigid pendulum (mass m_b and radius r) attached to the cup. Assuming planar xz-motion of the ball, we derive the Lagrange equations of motion [116] with three generalized coordinates $\mathbf{q}(t) = (x^{\text{cup}}(t), z^{\text{cup}}(t), \phi(t))$, which denote the x position of the cup, z position of the cup, and swing angle of the ball with respect to the plumb line of the cup respectively at any time $t \geq 0$. We reduce the equations to the general *nominal* form

$$M(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + C(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) + G(\mathbf{q}(t)) = F(t), \ \forall t \geq 0, \tag{9.1}$$

where $M(\mathbf{q}(t))$ is the inertia matrix, $C(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ is the Coriolis matrix, $G(\mathbf{q}(t))$ is the gravity matrix, and $F(t)$ is the external input force at time t. Here $\dot{\mathbf{q}}(t)$ denotes the velocity of the cup and the angular velocity of the ball, and $\ddot{\mathbf{q}}(t)$ denotes the acceleration of the cup and the angular acceleration of the ball at any time $t \geq 0$. System (9.1) in state-space form is

$$\dot{\bar{x}}(t) = f(\bar{x}(t), F(t)), \tag{9.2}$$

where nominal state $\bar{x}(t) = [\mathbf{q}^\top(t), \dot{\mathbf{q}}^\top(t)]^\top \in \mathbb{R}^6$ for all time $t \geq 0$.

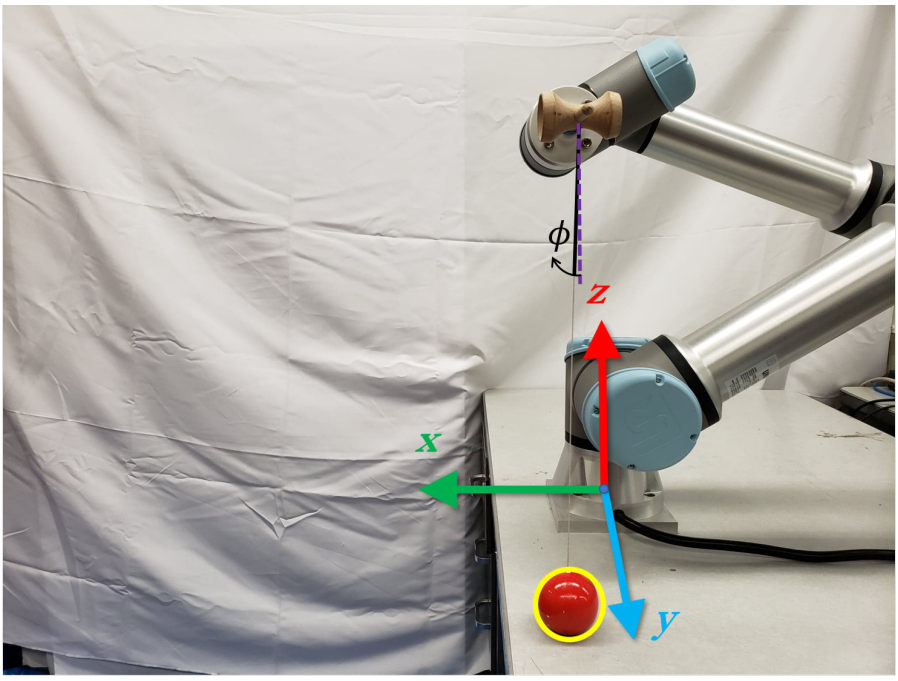


Figure 9.1: Manipulator with Kendama along with coordinate frame.

9.2.2 Optimization Problem

We discretize system (9.2) with one step Euler discretization and a sampling time of $T_s =$ 100Hz. The discrete time system can then be written as

$$\bar{x}_{i+1} = \bar{x}_i + T_s f(\bar{x}_i, F_i) = f_d(\bar{x}_i, F_i), \ \forall i \in \{0, 1, \dots\},$$

where a_i denotes the sampled time version of continuous variable $a(t)$. To generate a force input sequence for the swing-up, we solve a constrained optimal control problem over a finite planning horizon of length N, given by:

$$\begin{aligned} \min_{F_0,\dots,F_{N-1}} \quad & \sum_{i=0}^{N-1} \bar{x}_i^\top Q_s \bar{x}_i + F_i^\top R_s F_i \\ \text{s.t.,} \quad & \bar{x}_{i+1} = f_d(\bar{x}_i, F_i), \\ & \bar{x}_i \in \mathcal{X}, F_i \in \mathcal{F}, \\ & \bar{x}_0 = x_{\text{init}}, \\ & \bar{x}_N = x_{\text{f}}, \ i = 0, 1, \dots, (N-1), \end{aligned} \tag{9.3}$$

where weight matrices $Q_s, R_s \succ 0$, and constraint set $\mathcal{X}$ is chosen such that the ball remains within the reach of the UR5e manipulator. Initial state x_{init} is known in the configuration as shown in Fig. 9.1. Due to the nonlinear dynamics $f_d(\cdot, \cdot)$, the optimization problem (9.3) is non-convex. Moreover, typically a long horizon length N is required. Hence, we solve (9.3) *offline* and apply the computed input sequence $\mathbf{F}^\star = [F_0^\star, F_1^\star, \dots, F_{N-1}^\star]$ in open-loop to the manipulator.

9.2.3 Terminal Conditions of the Swing-Up

Predicted Behaviour

The nominal terminal state x_{f} in (9.3) is selected such that the ball is swinging to $\phi = 2.44$ rad with an angular velocity of $\dot{\phi} = 4.18$ rad/s. At these values, the string is calculated to lose tension and the ball begins free-fall. The chosen value of x_{f} ensures that the predicted difference in positions of the ball and the cup (both modeled as point masses) vanishes at a future time, if the cup were held fixed and the ball's motion is predicted under free-fall.

Actual Behaviour

When considering the nominal system (9.1), we have ignored the presence of uncertainties. Such uncertainties may arise due to our simplifying assumptions such as: (i) the string is mass-less so the swing angle is only affected by the ball and cup masses, (ii) there are no frictional and aerodynamic drag forces to hinder the conservation of kinetic and potential energy of the system, (iii) the cup mass is decoupled from the mass of the manipulator, and (iv) there is no mismatch of control commands from the low level controller of the manipulator and F. Due to such uncertainties, realized states x_i for $i \in \{0, 1, \ldots, N\}$ do not *exactly* match their nominal counterparts.

A set of 100 measured roll-out trajectories of the ball after the swing-up are shown in Fig. 9.2 for a fixed open-loop input sequence $\mathbf{F}^\star$. We see from Fig. 9.2 that after N timesteps

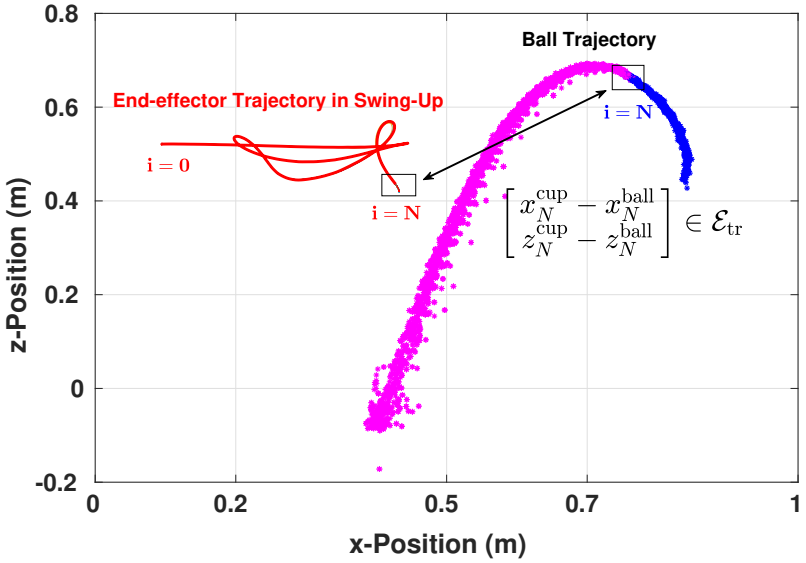


Figure 9.2: Start of catch phase (i.e., $i = N$) for 100 trajectories. Red line indicates the trajectory of the cup/end-effector during swing-up. Blue dots indicate ball positions during swing-up and pink dots indicate a position after catch phase is started. Closed-loop control begins when the relative position is in $\mathcal{E}_{\mathrm{tr}}$.

of swing-up, the ball and the cup arrive at positions where their relative position is in a set

$\mathcal{E}_{\text{tr}}$. A key assumption of well posedness will be imposed on this set in Section 9.3.4 in order for our subsequent feedback control policy to deliver a catch in experiments.

9.3 Designing Feedback Policy In Catch Phase

For the catch phase we start the time index $t = 0$ where the swing up ends, i.e., $i = N$. There are two main challenges during the design of the feedback controller, namely (i) position measurements of the ball from a noisy camera, and (ii) presence of mismatch between desired control actions and corresponding low level controller commands.

Assumption 9.1 *We assume that the UR5e end-effector gives an accurate estimate of its own position. The assumption is based on precision ranges provided in [***eseries***].*

9.3.1 Problem Formulation

During free-fall of the ball we design our feedback controller for the manipulator position *only* in end-effector space, with desired velocity of the end-effector as our control input. The joint ball and end-effector system in one trial can be modeled as a single integrator as:

$$e_{t+1} = Ae_t + Bu_t + w_t(e_t, u_t), \tag{9.4a}$$
$$y_t = e_t + v_t, \tag{9.4b}$$

with error states and inputs (i.e., relative position and velocity)

$$e_t = \begin{bmatrix} x_t^{\text{cup}} - x_t^{\text{ball}} \\ z_t^{\text{cup}} - z_t^{\text{ball}} \end{bmatrix}, \; u_t = \begin{bmatrix} v_{x,t}^{\text{cup}} - v_{x,t}^{\text{ball}} \\ v_{z,t}^{\text{cup}} - v_{z,t}^{\text{ball}} \end{bmatrix},$$

where $w_t(e_t, u_t) \in \mathbb{W}_m \subset \mathbb{R}^2$, is a bounded uncertainty which arises due to the discrepancy between (i) the predicted and the actual velocity of the ball at any given timestep[1], and (ii) the commanded and the realized velocities of the end-effector, primarily due to the low level controller delays and limitations. System dynamics matrices $A = I_2$ and $B = dt \cdot I_2$ are known, and sampling time $dt = 0.01$ second. We assume an outer approximation $\mathbb{W}$ to the set $\mathbb{W}_m$, i.e., $\mathbb{W}_m \subseteq \mathbb{W}$ is known, and is a polytope. We consider noisy measurements of states due to the noise in camera position measurements, corrupted by $v_t \overset{\text{i.i.d.}}{\sim} \mathcal{P}$, with $\text{Supp}(\mathcal{P}) = \mathbb{V}$, where $\text{Supp}(\cdot)$ denotes the support of a distribution. We assume $\mathbb{V}$ is *not* exactly known.

Using the set $\mathcal{E}_{\text{tr}}$ (see Fig. 9.2), a set $\mathcal{E}$ containing the origin where the string is not taut and (9.4) is valid can then be chosen. We choose:

$$\gamma^{(i)} = \|\text{vert}^{(i)}(\mathcal{E}_{\text{tr}})\|_\infty, \; i \in \{1, 2\},$$
$$\mathcal{E} = \{x : -\gamma \leq x \leq \gamma\}, \; \gamma = [\gamma^{(1)}, \gamma^{(2)}]^\top, \tag{9.5}$$

[1] we use the camera position information for ball's velocity estimation

where $\text{vert}^{(i)}(\mathcal{A})$ denotes i^{th} row of all the vertices of the polytope $\mathcal{A}$, and as introduced previously, $\|\cdot\|$ denotes the vector norm. This ensures

$$e_0 \in \mathcal{E}_{\text{tr}} \implies e_0 \in \mathcal{E}. \tag{9.6}$$

As (9.6) holds true, we impose state and input constraints for all timesteps $t \geq 0$ as given by:

$$e_t \in \mathcal{E},\ u_t \in \mathcal{U}, \tag{9.7}$$

where set $\mathcal{U}$ is a polytope. We formulate the following finite horizon robust optimal control problem for feedback control design:

$$\begin{aligned} \min_{u_0, u_1(\cdot), \ldots} \quad & \sum_{t=0}^{T-1} \ell\left(\bar{e}_t, u_t\left(\bar{e}_t\right)\right) + Q(\bar{e}_T) \\ \text{s.t.,} \quad & e_{t+1} = Ae_t + Bu_t(e_t) + w_t(e_t, u_t), \\ & \bar{e}_{t+1} = A\bar{e}_t + Bu_t(\bar{e}_t), \\ & y_t = e_t + v_t, \\ & e_t \in \mathcal{E}, u_t(e_t) \in \mathcal{U}, \\ & \forall w_t(e_t, u_t) \in \mathbb{W},\ \forall v_t \in \mathbb{V}, \\ & e_0 \in \mathcal{E},\ t = 0, 1, \ldots, (T-1), \end{aligned} \tag{9.8}$$

where e_t, u_t and $w_t(e_t, u_t)$ denote the realized system state, control input and model uncertainty at timestep t respectively, and $(\bar{e}_t, u_t(\bar{e}_t))$ denote the nominal state and corresponding nominal input. Notice that (9.8) minimizes the nominal cost over a task duration of length T decided by the user, having considered the safety restrictions during an experiment. The cost comprises of the positive definite stage cost $\ell(\cdot, \cdot)$, and the terminal cost $Q(\cdot)$.

The main challenge in solving problem (9.8) is that it is difficult to obtain the camera measurement noise distribution support $\mathbb{V}$. Resorting to worst-case a-priori set estimates of $\mathbb{V}$ as in [22, 13] might result in loss of feasibility of (9.8). To avoid this, we use a data-driven estimate of $\mathbb{V}$ denoted by $\hat{\mathbb{V}}(n)$, where n is the number of samples of noise v_t used to construct the set.

9.3.2 Control Formulation

As we have noisy output feedback in (9.12), we follow [46] for a tractable constrained finite time optimal controller design strategy. We repeatedly solve (9.8) at times $0 \leq t \leq (T-1)$ in a shrinking horizon fashion [6, Chapter 9]. We make the following assumption for this purpose:

Assumption 9.2 *The sets $\mathbb{W}_m, \mathbb{W}, \mathbb{V}$, and $\mathcal{U}$ contain the origin in their interior.*

Observer Design and Control Policy Parametrization

We design a Luenberger observer for the state as

$$\hat{e}_{t+1} = A\hat{e}_t + Bu_t + L(y_t - \hat{e}_t),$$

where the observer gain L is chosen such that $(A - L)$ is Schur stable. The control policy parametrization for solving (9.8) is chosen as:

$$u_t = \bar{u}_t + K(\hat{e}_t - \bar{e}_t),$$

where state feedback policy gain matrix K is chosen such that $(A + BK)$ is Schur stable.

Optimal Control Problem

Consider the tightened constraint sets,

$$\bar{\mathcal{E}}(n) = \mathcal{E} \ominus (\mathcal{R}^{\text{est}}(n) \oplus \mathcal{R}^{\text{con}}(n)), \tag{9.9a}$$
$$\bar{\mathcal{U}}(n) = \mathcal{U} \ominus K\mathcal{R}^{\text{con}}(n), \tag{9.9b}$$

where following [46, Proposition 1-2], the set $\mathcal{R}^{\text{est}}(n)$ is our best estimate of the minimal Robust Positive Invariant set $\mathcal{R}^{\text{est}}$ for *estimation* error $\delta e_t^{\text{est}} = e_t - \hat{e}_t$ dynamics defined as

$$\delta e_{t+1}^{\text{est}} = (A - L)\delta e_t^{\text{est}} + w_t(e_t, u_t) - Lv_t, \tag{9.10}$$

and the set $\mathcal{R}^{\text{con}}(n)$ is our best estimate of the minimal Robust Positive Invariant set $\mathcal{R}^{\text{con}}$ for the *control* error $\delta e_t^{\text{con}} = \hat{e}_t - \bar{e}_t$ dynamics defined as

$$\delta e_{t+1}^{\text{con}} = (A + BK)\delta e_t^{\text{con}} + L\delta e_t^{\text{est}} + Lv_t, \tag{9.11}$$

with $v_t \in \hat{\mathbb{V}}(n)$ and $w_t(e_t, u_t) \in \mathbb{W}$. We use the phrase *best estimate* for the above sets, since $\hat{\mathbb{V}}(n)$ is an estimate of true and unknown set $\mathbb{V}$.

Using these sets we then solve the following tractable finite horizon constrained optimal control problem at any timestep $t \geq 0$ as an approximation to (9.8):

$$\begin{aligned}
&V_{t\to T}^{\star}(\bar{\mathcal{E}}(n), \bar{\mathcal{U}}(n), \mathcal{R}^{\text{con}}(n), \hat{e}_t) := \\
&\min_{\bar{e}_t, \bar{u}_t, \ldots, \bar{u}_{T-1}} \quad \sum_{k=t}^{T-1} \ell(\bar{e}_k, \bar{u}_k) + Q(\bar{e}_T) \\
&\text{s.t.,} \qquad \bar{e}_{k+1} = A\bar{e}_k + B\bar{u}_k, \\
&\qquad\qquad u_k = \bar{u}_k + K(\hat{e}_k - \bar{e}_k), \\
&\qquad\qquad \bar{e}_k \in \bar{\mathcal{E}}(n), \bar{u}_k \in \bar{\mathcal{U}}(n), \\
&\qquad\qquad \hat{e}_t - \bar{e}_t \in \mathcal{R}^{\text{con}}(n), \\
&\qquad\qquad \bar{e}_T = 0, \\
&\qquad\qquad \forall k \in \{t, t+1, \ldots, (T-1)\},
\end{aligned} \tag{9.12}$$

where $\hat{e}_t$ is the observed state at timestep t, and $\{\bar{e}_k, \bar{u}_k\}$ denote the nominal state and corresponding input respectively predicted at timestep $k \geq t$. After solving (9.12), in closed-loop we apply

$$u_t^\star(e_t) : u_t^\star = \bar{u}_t^\star + K(\hat{e}_t - \bar{e}_t^\star) \tag{9.13}$$

to system (9.4). We then resolve the problem (9.12) again at the next $(t+1)$-th timestep, yielding a shrinking horizon strategy. The choice of initial observer state is made as follows:

$$\hat{e}_0 \in -(\mathcal{R}^{\text{est}}(n) \ominus \mathcal{E}). \tag{9.14}$$

Assumption 9.3 (Manipulator Speed) *If any feasible solution is found to* (9.12) *satisfying velocity error constraints* $\bar{\mathcal{U}}(n)$*, the manipulator has enough velocity authority to satisfy these constraints, where the predicted ball velocity is obtained using forward Euler integration at free-fall.*

Recall the set $\mathcal{E}_{\text{tr}}$ containing the set of all possible errors e_0 at the start of the catch phase, shown in Fig. 9.2. We now make the following assumption.

Assumption 9.4 (Well Posedness) *We assume that given state* $e_0 \in \mathcal{E}_{\text{tr}}$*, optimization problem* (9.12) *is feasible at all timesteps* $0 \leq t \leq (T-1)$ *with model uncertainty support* $\mathbb{W}$*, and true measurement noise support* $\hat{\mathbb{V}}(n) = \mathbb{V}$ *used in* (9.10)-(9.11) *and* (9.14)*, when* (9.13) *is applied to* (9.4) *in closed-loop. This implies that* $e_t \in \mathcal{E}$ *for all* $0 \leq t \leq T$*, where* $\mathcal{E}$ *is obtained from* $\mathcal{E}_{\text{tr}}$ *following* (9.5)*.*

Definition 9.1 (Trial Failure) *A Trial Failure at timestep* t *is the event*

$$[\text{TF}]_t : \ e_t \notin \mathcal{E}, \ 0 \leq t \leq T.$$

That is, a Trial Failure implies the violation of imposed constraints (9.7) *by system* (9.4) *in closed-loop with feedback controller* (9.13)*.*

Note that a Trial Failure is a possible scenario only because $\mathbb{V}$ is unknown and is estimated with $\hat{\mathbb{V}}(n)$ in (9.12). Intuitively, a Trial Failure implies one of the following:

(P1) Problem (9.12) losing feasibility during $0 < t < T$. This happens if $\hat{\mathbb{V}}(n) \not\supset \mathbb{V}$.

(P2) Problem (9.12) losing feasibility initially at $t = 0$, and/or sets $\bar{\mathcal{E}}(n)$, $\bar{\mathcal{U}}(n)$ becoming empty. This can happen if $\hat{\mathbb{V}}(n) \supset \mathbb{V}$.

9.3.3 Constructing Set $\hat{\mathbb{V}}(n)$

As described in Section 9.3.1 the set $\hat{\mathbb{V}}(n)$ is an estimate of the measurement noise support $\mathbb{V}$, derived from n samples of noise v_t. The set $\hat{\mathbb{V}}(n)$ is then used to compute $\mathcal{R}^{\text{est}}(n)$ and $\mathcal{R}^{\text{con}}(n)$ in (9.10)-(9.11), used in (9.12) and (9.14). We consider the following two design specifications while constructing set $\hat{\mathbb{V}}(n)$, given a fixed sample size n.

(D1) Probability of the event $\hat{\mathbb{V}}(n) \not\supset \mathbb{V}$ is bounded with a user specified upper bound ϵ.

(D2) Estimate $\hat{\mathbb{V}}(n)$ ensures event (P2) in Trial Failure occurs with a vanishing probability, while satisfying specification (D1).

Satisfying (D1) using Distribution Information

Fig. 9.1 shows the configuration of the system when n noise samples are collected to construct $\hat{\mathbb{V}}(n)$. Let Assumption 9.1 hold true and the ball is held still, vertically below the end-effector at a position, whose z-coordinate $z^{\text{cup}} = \bar{z}$ is fixed and *known* from previous UR5e end-effector measurements, and x-coordinate is fixed at $x^{\text{ball}} = 0$. We then collect n camera position measurements of the ball at this configuration. The discrepancy between the known position and the measurements yield values of noise samples $\mathbf{v}_n = [v_0, v_1, \ldots, v_n]$. For a fixed *environment*,[2] the distribution of collected samples is shown in Fig. 9.3, which is approximately a truncated normal distribution. We thereby consider this distribution family

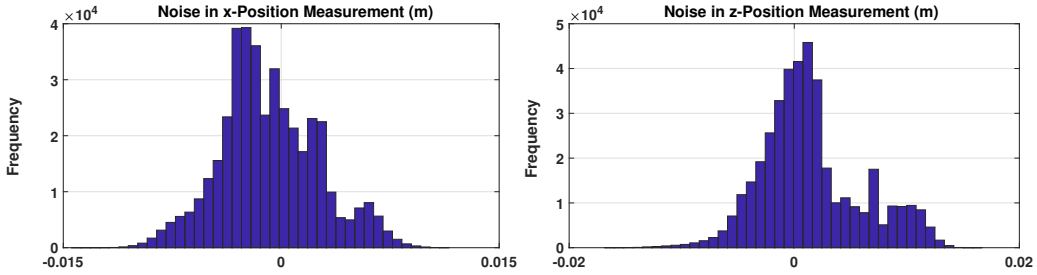


Figure 9.3: Camera measurement noise distribution histogram for a fixed camera environment using $n = 400,000$ samples.

in Fig. 9.3 conditioned on any environment as

$$\mathcal{P}^q_{\theta_q}|\text{env} = \mathcal{N}_{\text{trunc}}(\mu_q, \sigma_q^2, 3), \text{ with } q \in \{1, 2\}, \tag{9.15}$$

where $\mathcal{P}_\theta$ denotes that the distribution $\mathcal{P}$ belongs to a parametric family (truncated normal) parametrized by $\theta = (\mu, \sigma)$, q denotes the q^{th} dimension (x and z directions), and parameters (μ_q, σ_q) are unknown. For a parametric distribution such as (9.15), for any chosen $\epsilon \in (0, 1)$, set $\hat{\mathbb{V}}(n)$ is then constructed as the $(1-\epsilon)$-Confidence Support of $\mathcal{P}_\theta|\text{env}$ using the method proposed in Chapter 7, which ensures

$$\mathbb{P}(\hat{\mathbb{V}}(n) \not\supset \mathbb{V}) \leq \epsilon. \tag{9.16}$$

Note that (9.16) is a sufficient condition to guarantee that if (D2) holds, solving (9.12) and applying (9.13) to (9.4) gives

$$\mathbb{P}(e_t \notin \mathcal{E}) \leq \epsilon, \ 0 \leq t \leq T, \tag{9.17}$$

if $\hat{\mathbb{V}}(n)$ is used to construct sets $\mathcal{R}^{\text{est}}(n)$ and $\mathcal{R}^{\text{con}}(n)$.

[2] camera environment is parametrized by factors such as lighting conditions, camera field of view, etc.

Satisfying (D2) using Assumption 9.4

Since Assumption 9.4 holds, there exists a number of noise samples n_ϵ for any $\epsilon \in (0,1)$, such that $\hat{\mathbb{V}}(n_\epsilon)$ satisfies (D2). Thus, only the sample size n has to be chosen[3] for $\hat{\mathbb{V}}(n)$ appropriately to satisfy (D2), having ensured (9.17). This guarantees that constructing sets $\mathcal{R}^{\text{con}}(n)$ and $\mathcal{R}^{\text{est}}(n)$ using $\hat{\mathbb{V}}(n)$ and then designing a feedback control by solving (9.12) results in problem (9.12) being feasible throughout the task with probability at least $\beta = (1-\epsilon)^{T-1}$. Value of ϵ can be chosen small enough for any user-specified level β can be attained.

9.3.4 Obtaining Successful Catches

Constructing $\hat{\mathbb{V}}(n)$ as per Section 9.3.3 to ensure (9.17) is still *not* a sufficient condition to obtain a catch in an experiment with specified probability β, as our model (9.4) does not account for additional factors such as object dimensions, presence of contact forces, etc. Therefore we introduce the notion of a *successful catch*, which is defined as the ball successfully ending up inside the cup at the end of a roll-out. Thus, a successful catch accounts for the dimensions of the ball and the cup, and the presence of contact forces.

Assumption 9.5 (Existence of a Successful Catch) *We assume that given an initial state $e_0 \in \mathcal{E}_{\text{tr}}$, an input policy obtained by solving (9.12) can yield a successful catch, if true measurement noise support $\mathbb{V}$ were known* exactly.

Remark 9.1 *From Chapter 7 we know that as long as confidence intervals for parameters (μ, σ) in (9.15) converge, $\hat{\mathbb{V}}(n) \to \mathbb{V}$ as $n \to \infty$. So, if sample size n is increased iteratively approaching $n \to \infty$, obtaining a successful catch guaranteed owing to Assumption 9.5. However if a precise positioning system like Vicon is used to collect the noise samples, due to limited access to such environments, collecting more samples and increasing n could be expensive. We therefore stick to our method of constructing $\hat{\mathbb{V}}(n)$ for a* fixed *n as per Section 9.3.3, and we attempt successful catches with multiple roll-outs by solving (9.12). For improving the empirical probability of successful catches in these roll-outs, one may then increase n and thus update the control policy. We demonstrate this in Section 9.4.2 and Section 9.4.3.*

9.4 Experimental Results

We present our preliminary experimental findings in this section. For our experiments, the original Kendama handle was modified to be attached to a 3D printed mount on the UR5e end-effector, as shown in Fig. 9.1. A single Intel RealSense D435 depth camera running at 60 FPS was used to estimate the position and velocity of the ball. The source code is available at https://github.com /monimoyb/kendama.

[3] for n fixed, ϵ can be increased while constructing $\hat{\mathbb{V}}(n)$ to satisfy (D2).

9.4.1 Control Design in the Catch Phase

Once the swing-up controller is designed as per Section 9.2.2 and an open-loop swing-up control sequence is applied to the manipulator, we design the feedback controller by finding approximate solutions to the following problem:

$$\begin{aligned}
&\min_{u_0,u_1(\cdot),\ldots} \sum_{t=0}^{T-1} 500\,\|\bar{e}_t\|_2^2 + 0.4\,\|u_t(\bar{e}_t)\|_2^2 \\
&\text{s.t.,} \\
&\qquad e_{t+1} = Ae_t + Bu_t(e_t), \\
&\qquad \bar{e}_{t+1} = A\bar{e}_t + Bu_t(\bar{e}_t), \\
&\qquad y_t = e_t + v_t, \\
&\qquad e_t \in \mathcal{E},\ \begin{bmatrix} -8\text{m/s} \\ -8\text{m/s} \end{bmatrix} \le u_t(e_t) \le \begin{bmatrix} 8\text{m/s} \\ 8\text{m/s} \end{bmatrix}, \\
&\qquad \forall v_t \in \mathbb{V}, \\
&\qquad t = 0, 1, \ldots, (T-1),
\end{aligned} \tag{9.18}$$

where set $\mathcal{E}_{\text{tr}} = [-0.316\text{m}, 0.349\text{m}] \times [-0.2095\text{m}, 0.2457\text{m}]$, shown in Fig. 9.2. Note that for this specific scenario the presence of model uncertainty can be ignored. Set $\mathbb{V}$ is unknown, and we consider Assumption 9.4 holds. System matrices A, B are from Section 9.3.1.

9.4.2 Mujoco: Increasing Successful Catches

We first conduct exhaustive Mujoco [117, 118] simulations of the catching problem, having formed $\hat{\mathbb{V}}(n)$ as per Section 9.3.3, with $n = 100$ and then iteratively increasing to $n = 2000$. Sets $\hat{\mathbb{V}}(n)$ are formed using the tools presented in Chapter 7. The task duration in this case is $T = 25$ steps. The trend in the percentage of successful catches with 1000 roll-outs corresponding to each n, varying from $n = 50$ to $n = 2000$, is shown in Fig. 9.4. For $n = 50$, 46.9% of the roll-outs result in a successful catch. The number increases to 68.3% for $n = 2000$. Thus we prove that our proposed approach enables successful learning of the kendama ball catching task in high-fidelity Mujoco simulations. Note that although the percentage of successful catches demonstrably improves, but it does not attain a perfect 100% value. This is due to the simulated nonlinearities in Mujoco at the time of impact, which are not captured by our linear model (9.4) in the catching phase.

9.4.3 UR5e: Learning to Catch

In order to validate the findings from Mujoco simulations, we conduct 50 *roll-outs* of the catching task on our experimental UR5e robot by solving (9.12). Fig. 9.5 shows the percentage of roll-outs conducted for each iteration (i.e., for each value of n), that resulted in the ball successfully striking the center of the cup. The percentage increases from 41.46% to 61.62%. Furthermore, another crucial quantity at the time of impact is the commanded

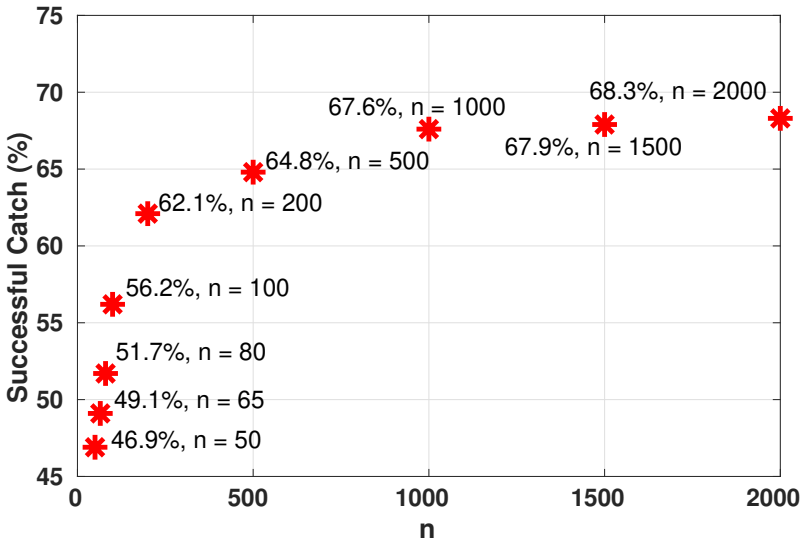


Figure 9.4: Percentage of successful catches vs sample size n.

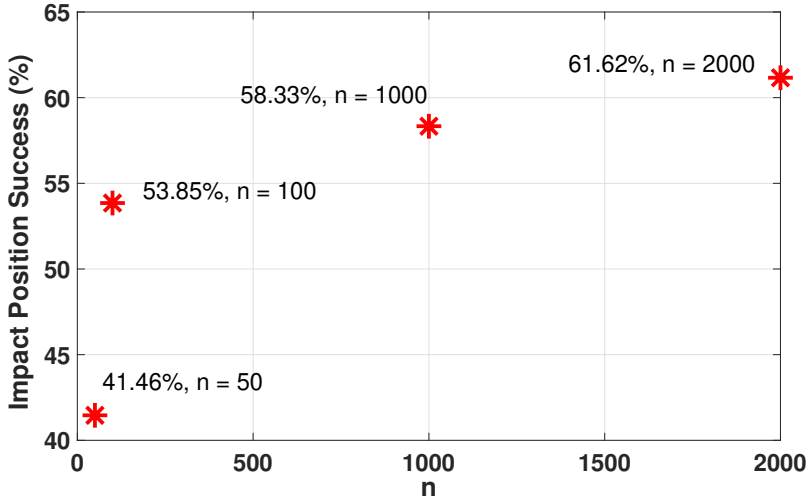


Figure 9.5: Percentage of times the ball hitting the cup center among all roll-outs vs sample size n.

relative velocity (9.13) in z-direction, a lower value of which indicates an increased likelihood of the ball not bouncing out. The average value and the standard deviation of of $(u^{\star}_{T_{\rm im}-1})^{*\tilde{m}}_{z}$ for $\tilde{m} \in \{1, 2, \ldots, 50\}$ is shown in Fig. 9.6, where $(\cdot)^{*\tilde{m}}$ denotes the $\tilde{m}^{\rm th}$ roll-out and $T_{\rm im} \leq T$ denotes the time of impact. As seen in Fig. 9.6, the mean of the relative velocity at impact lowers from 0.38 m/s to -0.06 m/s. This together with Fig. 9.5 indicates a possibility of increasing successful catch counts as n is increased. As a matter of fact, at the end of this

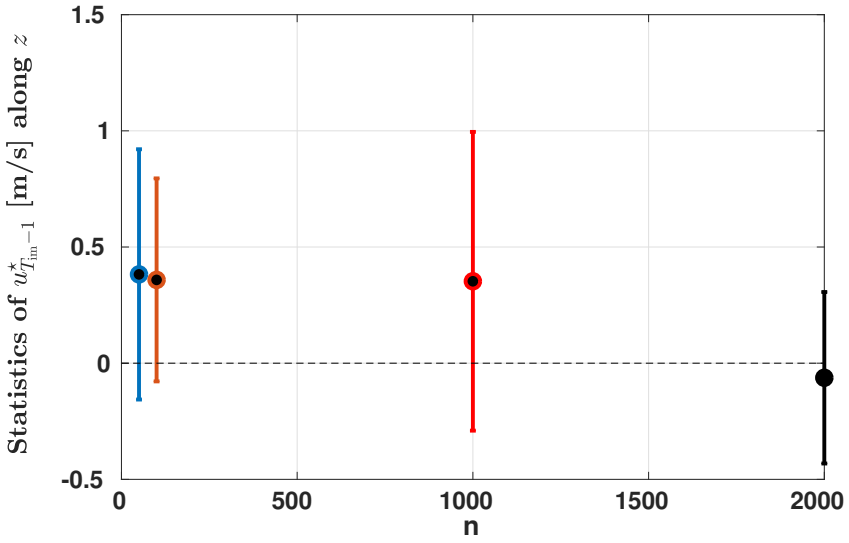


Figure 9.6: One standard deviation interval around the mean (circle) of z-relative velocity at impact, i.e., $[u^{\star}_{T_{\rm im}-1}]_z$ vs sample size n.

learning process, the robot learns to catch the ball successfully, and repeatably. An example case is demonstrated in: video link. However, a noticeable difference is seen in this case between the Trial Failure probability and the rate of failed catches. Recall their difference from Section 9.3.4. We discuss this in the next section.

9.4.4 Catch Percentage in Experiments

The percentage of successful catches by the robot during experiments is found to be lower than the values obtained in Fig. 9.4. We hypothesize that this is due to the presence of un-modeled factors in our simplified system model (9.4), such as the vibrations of the table and the robot arm, contact dynamics, low-level controller delays/saturation, etc. Therefore, an informative extension would be to collect the ball's position information at impact during successful catches using a position tracking unit such as Vicon, and compare the corresponding set of error values, denoted by $\mathcal{E}^{\rm tr}_{\rm catch}$ with the set $\mathcal{R}^{\rm est}(n) \oplus \mathcal{R}^{\rm con}(n)$ for an $n \gg 0$. A set $\mathcal{E}^{\rm tr}_{\rm catch} \subset \mathcal{R}^{\rm est}(n) \oplus \mathcal{R}^{\rm con}(n)$ with:

$$\text{volume}(\mathcal{E}^{\rm tr}_{\rm catch}) \ll \text{volume}(\mathcal{R}^{\rm est}(n) \oplus \mathcal{R}^{\rm con}(n))$$

would be demonstrably suggestive of the limitations of linear modeling used in this chapter. We leave such a validation and potential policy improvements as a future extension of the presented work in this Chapter.

9.5 Chapter Summary

We proposed a model based control strategy for the classic cup-and-ball game. The control problem was divided into two sub-tasks, for swinging the ball up, and then catching the free-falling ball, respectively. The swing-up trajectory ensures that a successful catch is possible with our feedback control design approach. Subsequently, a convex optimization problem was solved online during the ball's free-fall to control the manipulator and catch the ball. The controller utilized noisy position measurements of the ball from a camera, and the support of this noise distribution was iteratively learned from data using the algorithm from Chapter 7. Thus, the closed-loop control policy iteratively updates. We theoretically proved that under the considered modeling assumptions, the probability of a catch increases in the limit. Preliminary experimental results and high-fidelity simulations support our analysis.

Chapter 10

Decentralized Robotic Collaboration: An Application of Constraint Learning

This chapter is based on the published work in [43]. Obstacle avoidance in collaborative robotics has primarily considered known obstacles and solving a centralized problem with explicit communication [119, 120, 121, 122, 123, 124]. The use of implicit communication, such as force and torque measurements or estimates on rigid bodies [125, 126, 127, 128, 129, 130, 131, 132, 133, 134] remains challenging for obstacle avoidance when the robots rely only on local controllers in unstructured and unknown or partially known environments. In this proposed work, we extended the notion of constraint learning presented in Chapter 8 to such robotic applications, where robots learn information of unknown obstacles in their proximity via the interaction information during collaboration.

10.1 Summary of Contributions

We consider the task of two robots collaboratively transporting an object, constraining the robots' inputs to comply with the object's physical constraints. We consider no explicit communication, so the local environment information and the control actions are not shared between the robots. We solve the control design problem by using a leader-follower strategy with the leader using an MPC and the follower using a simple controller, known to the leader. With this schema, the leader can solve the collaborative transportation task with the help of the follower, while building a map of its unknown obstacles. Such a map is obtained by estimating the follower's inputs to infer missing local information about the environment sensed by the follower. Our key contributions can thus be summarized as follows:

- We propose a leader-follower strategy for two robots collaboratively transporting an object in a partially known environment with obstacles. The leader solves an MPC problem based on its known set of obstacles and plans a trajectory to reach the target

position, while avoiding collisions for the whole system (i.e., the two robots combined with the object to be transported).

- We present a simple control policy for the follower that is reactive to obstacles detected by the follower (and possibly undetected by the leader). This follower control policy is designed so that it allows the leader to infer the position of obstacles not directly sensed.

- Motivated by [125], we introduce a strategy for allowing leader-follower role switches during the task. We present a detailed numerical example of two point robots transporting a rigid rod in an initially unknown environment. On this example our proposed approach allows the leader's MPC controller to learn the undetected obstacles and successfully complete the task, with the leader-follower roles appropriately switched.

Control design with three or more robots is not addressed in this work. In order to estimate the inputs of the other robot, we assume each robot can estimate the states of the joint system, i.e., the two robots with the object to be transported. For the considered example of two point robots transporting a rigid rod, this estimation is done with measurements of robots' own positions and the rod orientation. For more complicated systems, similar estimates may be obtained using additional sensors. We do not present this in this chapter.

10.2 Problem Formulation

In this section, we formulate the collaborative obstacle avoidance problem with the leader-follower control architecture. Such a leader-follower hierarchy is common in control design [135, 126, 136, 125]. We limit ourselves to the case of only one follower. We refer to the two robots with the object to be transported as the *joint system*.

10.2.1 Environment Constraints

Let the environment be defined by the set $\mathcal{X}$. We assume obstacles are static, although the proposed framework can be extended to dynamic obstacles. Let the set of obstacles be denoted by $\mathcal{O}$. Therefore, the safe set for the joint system is given by $\mathcal{S} = \mathcal{X} \setminus \mathcal{O}$. At the beginning of the task, we assume that the robots do not have any prior information about the environment. During the control task, the robots detect obstacles and store their positions. At any timestep t, let the set of obstacle constraints known to the leader and the follower (detected at t and stored until t) be denoted by $\mathcal{C}_{l,t}$ and $\mathcal{C}_{f,t}$, respectively. We denote:

$$\mathcal{C}_{l,t} \cup \mathcal{C}_{f,t} = \mathcal{O}_t, \text{ with } \mathcal{O}_{t-T_s} \subseteq \mathcal{O}_t, \ \forall t \leq T,$$

where T_s is the sampling time of both the leader and the follower robot controllers (defined next in Section 10.2.2), and $T \gg T_s$ is the task duration.

10.2.2 System Modeling

We consider that the leader and the follower robots transport the same object as they move. The state space equation of the joint system is of the form:

$$S_{t+T_s} = f(S_t, u_t, v_t), \tag{10.1}$$

where $S_t \in \mathbb{R}^n$ is the joint system state, $u_t \in \mathbb{R}^m$ is the input of the leader and $v_t \in \mathbb{R}^p$ is the input of the follower at timestep t, and $f(\cdot,\cdot,\cdot)$ is any nonlinear map.

Remark 10.1 *In general the states S_t contain the positions and velocities of the center of masses of the leader, the follower and the object being transported.*

A block diagram of the joint system is shown in Fig. 10.1, where the red and the blue parts indicate the operations carried out by the leader and the follower, respectively. We consider

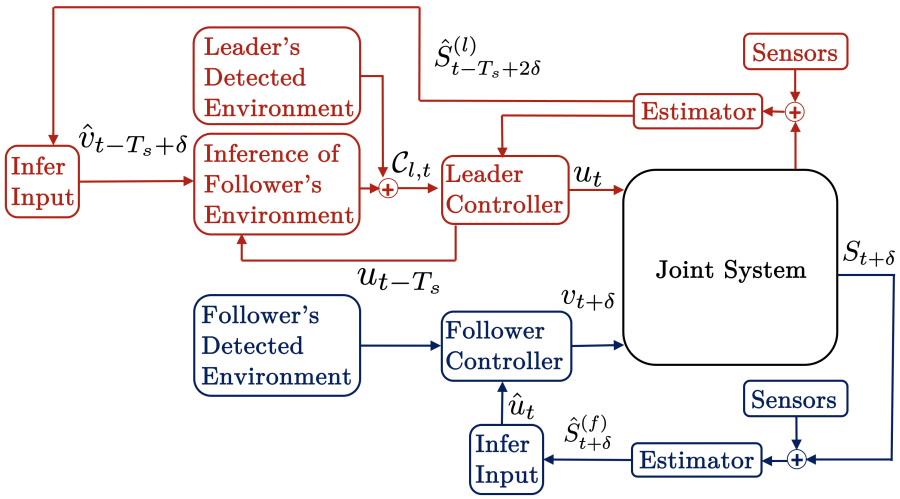


Figure 10.1: Block diagram of the joint system with leader follower controllers.

the case where the leader does not have full information of all the detected obstacles in $\mathcal{O}_t$, i.e., $\mathcal{C}_{l,t} \subset \mathcal{O}_t$. We further consider that no explicit communication between the leader and the follower is available. Similar to [125, 126, 127, 128, 129, 130, 131, 132, 133], we enable both the agents to infer each other's inputs as "implicit" communication. We first make the following assumption.

Assumption 10.1 *The leader and the follower can estimate the joint system states S_t at all timesteps.*

We introduce the following notation: let $(\cdot)^{(j)}_t$ denote the value of the quantity $(\cdot)_t$ as inferred by robot $j \in \{l, f\}$. The leader and the follower's estimates of S_t are thus denoted by $\hat{S}^{(l)}_t$ and $\hat{S}^{(f)}_t$, respectively. We denote the coordinates of the center of mass of the follower by

$R_t = [X_{f,t}, Y_{f,t}]^\top$, and the leader/follower estimates $\hat{R}_t^{(l/f)} = [\hat{X}_{f,t}^{(l/f)}, \hat{Y}_{f,t}^{(l/f)}]^\top$. Often such states are already included in $\hat{S}_t^{(l/f)}$, as pointed out in Remark 10.1. If they are not a part of $\hat{S}_t^{(l/f)}$, they need to be estimated as well in our control approach.

Method Outline: At timestep t, the leader uses $\hat{S}_t^{(l)}$ to compute the control action u_t for the joint system to avoid its known set of obstacles $\mathcal{C}_{l,t}$. As there is no explicit communication, the follower infers the leader's inputs u_t via its state estimates, inducing a delay in the application of its inputs. That is, at timestep $t+\delta$ (with a $\delta \ll T_s$), the follower uses $\hat{S}_{t+\delta}^{(f)}$ and $\hat{R}_{t+\delta}^{(f)}$ to infer $\hat{u}_t$. The follower also uses $\hat{R}_{t+\delta}^{(f)}$ to build a map of its detected obstacles, and computes $v_{t+\delta}$ as a function of u_t and these obstacles. During the inference time between t and $t+\delta$ the follower keeps applying the previous input $v_{(t-T_s)+\delta}$. The leader then infers the follower's inputs $v_{t+\delta}$ via its state estimates to learn additional obstacles. That is, at timestep $(t+2\delta)$[1], the leader uses $\hat{S}_{t+2\delta}^{(l)}$ to estimate $\hat{v}_{t+\delta}$, based on which it learns the position of any additional obstacles in the follower's proximity at $t+\delta$ using $\hat{R}_{t+\delta}^{(l)}$. The leader then computes updated $\mathcal{C}_{l,t+T_s}$. We detail the algorithm in Section 10.3. In the next sections, we present the controller synthesis. We discuss the effect of the time delay δ in details in Section 10.3.3-10.3.4, when we distinguish between the leader and the follower applied inputs.

Remark 10.2 *We consider that the leader and follower robots have synchronized clocks. A short discussion of non synchronized clocks is presented in Section 10.3.6.*

For clarity of presentation in this chapter, we present a specific case of model (10.1). Specifically, we model both the leader and the follower as point robots m_l and m_f, with global coordinates X_l, Y_l and X_f, Y_f, respectively, transporting a rigid rod, as shown in Fig. 10.2. The connecting rod of length $(l_l + l_f)$ has a mass m_r. Assuming the rod is of uniform density, the rod has an inertia, J_r, of $\frac{1}{12}m_r(l_l+l_f)^2$. The total mass of the joint system is therefore $m = m_r + m_l + m_f$. The total moment of inertia of the joint system is $J = J_r + m_r(\frac{l_l - l_f}{2})^2 + m_l l_l^2 + m_f l_f^2$. The leader's inputs on the rod are the axial force F_{al}, perpendicular force F_{pl}, and the torque τ_l. The corresponding follower's inputs are F_{af}, F_{pf} and τ_f. Denoting $\mathcal{T} = \frac{(-F_{pf}l_f + F_{pl}l_l + \tau_l + \tau_f)}{J}$ and define:

$$\begin{aligned} q_1 &= -(l_l \sin\theta \mathcal{T} + l_l \cos\theta \dot{\theta}^2) + \frac{1}{m}(\cos\theta(F_{al}+F_{af}) - \sin\theta(F_{pl}+F_{pf})), \\ q_2 &= (l_l \cos\theta \mathcal{T} - l_l \sin\theta \dot{\theta}^2) + \frac{1}{m}(\sin\theta(F_{al}+F_{af})) + \cos\theta(F_{pl}+F_{pf})). \end{aligned} \tag{10.2}$$

Due to the rigid coupling with the rod, the leader's position, and translational and angular velocity states are sufficient to define the evolution of the joint system. Accordingly, using (10.2), the state-space equation for the joint system is:

$$\dot{S}(t) = f_c(S(t), u(t), v(t)) = \begin{bmatrix} \dot{X}_l & q_1 & \dot{Y}_l & q_2 & \dot{\theta} & \mathcal{T} \end{bmatrix}^\top, \tag{10.3}$$

[1]the leader's inference can be made at timestep $(t+\delta+\mu)$ with $\delta < \mu \leq T_s$. We use $\delta \ll T_s$ and $\mu = \delta$ only to simplify notations. See Table 10.1.

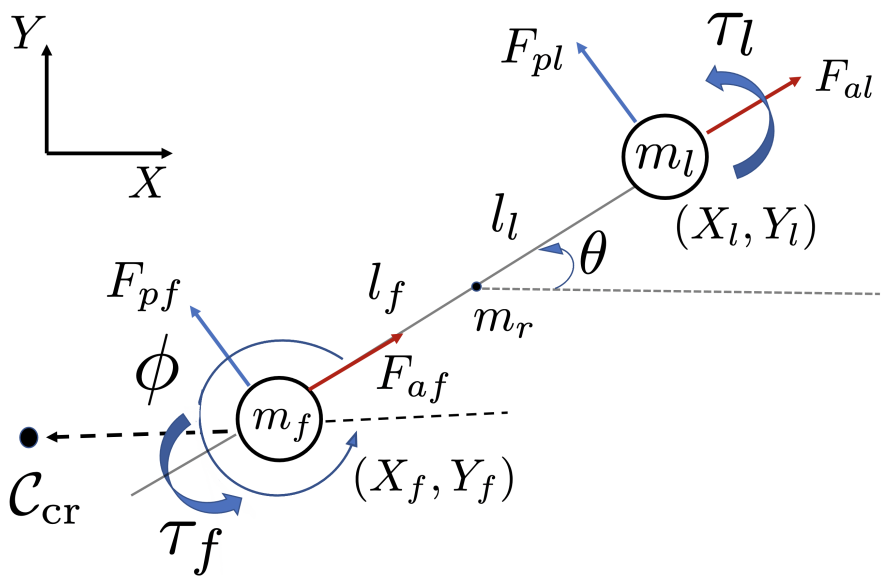


Figure 10.2: Model of the joint system. The leader and the follower are point masses connected by the rigid rod. The follower reacts to a *critical obstacle* $\mathcal{C}_{\text{cr}}$, as defined in Section 10.3.1.

with $S(t) = [s_1, s_2, s_3, s_4, s_5, s_6]^\top$, $u(t) = [F_{al}, F_{pl}, \tau_l]^\top$, and $v(t) = [F_{af}, F_{pf}, \tau_f]^\top$ at time t, where the states are representative of variables given by:

$$s_1 = X_l, s_2 = \dot{X}_l, s_3 = Y_l, s_4 = \dot{Y}_l, s_5 = \theta, s_6 = \dot{\theta}.$$

We discretize (10.3) with the sampling time of T_s for both the leader and the follower to obtain its discrete time version similar to (10.1). Furthermore, for this specific model (10.3), Assumption 10.1 can be stated as: both the leader and the follower can estimate the leader's position, velocity, as well as the angular speed and orientation of the rod at all times. For simplicity of presentation, we consider that the robots measure their positions, velocities, and the rod's angle and angular speed. Accordingly, estimators $\hat{S}_t^{(l)}$ and $\hat{S}_t^{(f)}$ are:

$$\hat{S}_t^{(l)} = \begin{bmatrix} X_{l,t} & \dot{X}_{l,t} & Y_{l,t} & \dot{Y}_{l,t} & \theta_t & \dot{\theta}_t \end{bmatrix}^\top, \tag{10.4a}$$

$$\hat{S}_t^{(f)} = \begin{bmatrix} \hat{X}_{l,t}^{(f)} \\ \hat{\dot{X}}_{l,t}^{(f)} \\ \hat{Y}_{l,t}^{(f)} \\ \hat{\dot{Y}}_{l,t}^{(f)} \\ \theta_t \\ \dot{\theta}_t \end{bmatrix} = \begin{bmatrix} X_{f,t} + (l_l + l_f)\cos\theta_t \\ \dot{X}_{f,t} - (l_l + l_f)\sin\theta_t\dot{\theta}_t \\ Y_{f,t} + (l_l + l_f)\sin\theta_t \\ \dot{Y}_{f,t} + (l_l + l_f)\cos\theta_t\dot{\theta}_t \\ \theta_t \\ \dot{\theta}_t \end{bmatrix}. \tag{10.4b}$$

In the absence of perfect position, velocity and rod orientation measurements, one can design appropriate state observers, such as a particle filter or an extended Kalman filter to obtain their estimates, if Assumption 10.1 holds.

10.2.3 Input Constraints

We consider constraints on the inputs of both the leader and the follower, which are given by $u_t \in \mathcal{U}$ and $v_t \in \mathcal{V}$ for all $t \geq 0$. For our specific example in this chapter, with $\bar{F}_a, \bar{F}_p, \bar{\tau} \in \mathbb{R}_+$, we consider the same box constraints:

$$\mathcal{U} = \mathcal{V} := \{w : -\begin{bmatrix}\bar{F}_a & \bar{F}_p & \bar{\tau}\end{bmatrix}^\top \leq w \leq \begin{bmatrix}\bar{F}_a & \bar{F}_p & \bar{\tau}\end{bmatrix}^\top\}. \tag{10.5}$$

10.3 Control Synthesis

We enumerate the steps involved in our leader-follower control synthesis briefly next, which constitute our *collaborative obstacle avoidance with environment learning* algorithm.

(I) At any timestep t, the leader designs an MPC controller with horizon of N steps with $NT_s \ll T$ for the joint system to reach a specified target position S_{tar}, while avoiding all the stored obstacles in $\mathcal{C}_{l,t}$. This is shown in Section 10.3.2.

(II) If there are no obstacles in its proximity, the follower uses a control strategy to support the actions of the leader. The inference of the leader actions by the follower is described in Section 10.3.3.

(III) In the case where critical obstacles (as defined later in Definition 10.1) are detected by the follower, the follower applies an additional input contribution in order to avoid these critical obstacles, as we show in Section 10.3.1.

(IV) The leader estimates the follower's applied inputs and uses this as an "implicit" communication to build a map of its possibly unseen obstacles lying in the follower's proximity. The leader then updates its set of known obstacles $\mathcal{C}_{l,t+T_s}$, as we show in Section 10.3.4. The leader MPC problem is solved again at the next timestep with the updated environment information.

We now elaborate the above steps (I)-(IV) in the following sections. The resulting *collaborative obstacle avoidance with environment learning* algorithm is in Section 10.3.6.

10.3.1 Follower Policy Parameterization

In the set of all obstacles seen by the follower, we define a *critical obstacle point*, due to which the follower chooses to apply a reactive input.

Definition 10.1 (Critical Obstacle Points) *We define a critical obstacle point at timestep t as a* point *in the set of obstacles $\mathcal{C}_{f,t}$ which is within a radius of d_{cr} from the follower's center of mass. Thus, the follower computes:*

$$\begin{aligned}\mathcal{C}_{\text{cr},t} = \arg\min_{c \in \mathcal{C}_{f,t}} \|\hat{R}_t^{(f)} - c\| \\ \text{s.t., } \|\hat{R}_t^{(f)} - c\| \leq d_{\text{cr}},\end{aligned} \tag{10.6}$$

where $\|\cdot\|$ *denotes the Euclidean norm.*

In case of multiple critical obstacle points satisfying (10.6), we pick the critical obstacle point as the one that maximizes

$$\max_{c\in\mathcal{C}_{\mathrm{cr},t}} \frac{\dot{\hat{R}}_t^{(f)}\cdot(c-\hat{R}_t^{(f)})}{\|\dot{\hat{R}}_t^{(f)}\|\|(c-\hat{R}_t^{(f)})\|},$$

that is, the one having the maximum relative velocity component towards the follower's center of mass. The inputs applied by the follower are then given by:

$$v_t=\begin{cases} f_1(u_t), & \text{if no critical obstacle point at } t,\\ f_2(u_t,d_{\mathrm{cr}},d_t,\phi_t) & \text{otherwise,}\end{cases} \tag{10.7}$$

where $f_1(\cdot)$ and $f_2(\cdot,\cdot,\cdot,\cdot)$ can be any function chosen such that $v_t\in\mathcal{V}$, u_t is the input of the leader, $d_t=\|\hat{R}_t^{(f)}-\mathcal{C}_{\mathrm{cr},t}\|$, ϕ_t is the angle between the vector connecting the follower center of mass to critical obstacle point and the follower center of mass to that of the leader, respectively. For our considered specific example, this is shown in Fig. 10.2.

We now make the following assumption ensuring when a critical obstacle point is seen, the follower applies a separate input, as opposed to what it would have applied otherwise.

Assumption 10.2 *We assume in* (10.7)*:*

$$\forall t\geq 0,\ \nexists\ u_t,d_t,\phi_t : d_t\leq d_{\mathrm{cr}},\ \ f_1(u_t)=f_2(u_t,d_{\mathrm{cr}},d_t,\phi_t).$$

We also make the following assumption that will be used for the leader's control synthesis in Section 10.3.2 and for learning critical obstacle points in Section 10.3.4.

Assumption 10.3 *We assume that the functions* $f_1(\cdot)$ *and* $f_2(\cdot,\cdot,\cdot,\cdot)$ *are known to the leader.*

Assumption 10.3 holds true, since such basic information can be shared offline before the task begins. Otherwise, these functions can be learned offline from data. Our specific choice of (10.7) in this chapter is given by:

$$v_t=\begin{cases} K_2u_t, & \text{if no critical obstacle point at } t,\\ K_2u_t+K_1(d_{\mathrm{cr}}-d_t)\begin{bmatrix}\cos\phi_t\\ -\sin\phi_t\\ 0\end{bmatrix} & \text{otherwise,}\end{cases} \tag{10.8}$$

where in d_t we directly measure R_t, i.e., $\hat{R}_t^{(f)}=R_t$ (see (10.4b)), and the gains K_1 and K_2 known to the leader, chosen to satisfy (10.5). Without any loss of generality in (10.8), we

have not included a reactive torque upon seeing critical obstacle points. Hence, only the first 2×2 sub-matrix of K_1 need to be invertible. We choose $K_2 \in [0, 1)$, and

$$K_1 = \text{diag}\Big(\frac{\bar{F}_a(1-K_2)}{d_{\text{cr}}}, \frac{\bar{F}_p(1-K_2)}{d_{\text{cr}}}, 0\Big),$$

ensuring the follower's inputs are saturated only at $d_t = 0$.

10.3.2 MPC Controller of the Leader

Using Assumption 10.1 and Assumption 10.3, the constrained finite time optimal control problem that the leader has to solve for its MPC controller synthesis at timestep t is:

$$\begin{aligned}
\min_{U_t} \ & \sum_{k=1}^{N} [(S_{t+kT_s|t} - S_{\text{tar}})^\top Q_s (S_{t+kT_s|t} - S_{\text{tar}}) + u_{t+(k-1)T_s|t}^\top Q_i u_{t+(k-1)T_s|t}] \\
\text{s.t.}, \ & S_{t+kT_s|t} = f(S_{t+(k-1)T_s|t}, u_{t+(k-1)T_s|t}, v_{t+(k-1)T_s|t}), \\
& \mathcal{B}(S_{t+kT_s|t}) \in \mathcal{X} \setminus \mathcal{C}_{l,t}, \\
& u_{t+(k-1)T_s|t} \in \mathcal{U}, \ v_{t+(k-1)T_s|t} = f_1(u_{t+(k-1)T_s|t}), \\
& \forall k \in \{1, 2, \ldots, N\}, \ S_{t|t} = \hat{S}_t^{(l)},
\end{aligned} \tag{10.9}$$

where $\mathcal{B}(\cdot)$ is a set of position coordinates defining the joint leader-follower system, input sequence $U_t = \{u_{t|t}, \ldots, u_{t+(N-1)T_s|t}\}$, S_{tar} is the target state, and $Q_s, Q_i \succcurlyeq 0$ are the weight matrices. Note, in order to avoid a mixed integer formulation arising due to all possible combinations of the follower's critical obstacle points in $\mathcal{C}_{l,t}$ along the prediction horizon, in (10.9) the leader computes the predicted $v_{k|t}$ using only $f_1(\cdot)$. For model (10.3) with follower policy (10.8), the leader uses (10.4a) to estimate:

$$R_{t+kT_s|t} = \begin{bmatrix} s_{1,t+kT_s|t} - (l_f + l_r)\cos s_{5,t+kT_s|t} \\ s_{3,t+kT_s|t} - (l_f + l_r)\sin s_{5,t+kT_s|t} \end{bmatrix}, \tag{10.10a}$$

$$\mathcal{B}(S_{t+kT_s|t}) = \{x : \exists \alpha \in [0,1], \ x = \alpha \begin{bmatrix} s_{1,t+kT_s|t} \\ s_{3,t+kT_s|t} \end{bmatrix} + (1-\alpha) R_{t+kT_s|t}\}, \tag{10.10b}$$

$$v_{t+(k-1)T_s|t} = K_2 u_{t+(k-1)T_s|t} \in \mathcal{U}, \ S_{t|t} = \hat{S}_t^{(l)}, \tag{10.10c}$$

in (10.9), for all $k \in \{1, 2, \ldots, N\}$, with $\mathcal{U}$ from (10.5). Solving (10.9)-(10.10) is difficult, mostly due to the non-convexity of the imposed state constraints $\mathcal{X} \setminus \mathcal{C}_{l,t}$, and that too for all values of parameter $\alpha \in [0, 1]$. Therefore, we solve an approximation to (10.9)-(10.10), as shown in the Appendix. After finding a solution to (10.9), the leader applies input

$$u_t = u_{t|t}^\star \tag{10.11}$$

to joint system (10.1) in closed-loop. Since the follower has no direct access to (10.11) to apply its own inputs according to (10.7), it estimates the leader's inputs. This is elaborated in the next section.

10.3.3 Applying the Follower's Inputs

The follower uses $\hat{S}_t^{(f)}$ to construct an estimate $\hat{u}_t$ of the leader's inputs u_t. This inference is done in a time duration of $\delta \ll T_s$ after timestep t, as introduced in Fig. 10.1 and Section 10.2.2. For this inference to be feasible, we make the following sufficient assumption. Let $\hat{S}_t^{(f)} \in \mathcal{Y}_f$, $\forall t \geq 0$.

Assumption 10.4 *We assume that the map from the set $\mathcal{U}$ to the set $\mathcal{Y}_f$ is invertible.*

Assumption 10.4 ensures that by using its set of estimates for the leader's states, the follower has the ability to uniquely infer the input u_t applied by the leader. Between the timesteps t and $t+\delta$ the follower applies its previous inputs $v_{t-T_s+\delta}$. Afterwards, the follower applies

$$v_{t+\delta} = \begin{cases} f_1(\hat{u}_t), & \text{if no critical obstacle point at } t+\delta, \\ f_2(\hat{u}_t, d_{\text{cr}}, d_{t+\delta}, \phi_{t+\delta}), & \text{otherwise,} \end{cases} \tag{10.12}$$

where the computation of $\hat{u}_t$ uses Assumptions 10.1 and 10.4. For our considered system model (10.3), the follower's estimates of the joint system states (i.e., the leader's states) are given in (10.4b). This satisfies Assumption 10.4. The construction of the estimate $\hat{u}_t$ and the corresponding form of the follower's applied inputs

$$v_{t+\delta} = \begin{cases} K_2\hat{u}_t, & \text{if no critical obstacle point at } t+\delta, \\ K_2\hat{u}_t + K_1(d_{\text{cr}} - d_{t+\delta}) \begin{bmatrix} \cos\phi_{t+\delta} \\ -\sin\phi_{t+\delta} \\ 0 \end{bmatrix} & \text{otherwise,} \end{cases} \tag{10.13}$$

where $d_{t+\delta} = \|R_{t+\delta} - \mathcal{C}_{\text{cr},t+\delta}\|$, are derived in detail in the Appendix. Here we directly measure $R_{t+\delta}$, i.e., $\hat{R}_{t+\delta}^{(f)} = R_{t+\delta}$ (see (10.4b)). Similar derivations can be conducted for variations of (10.3), e.g., the rigid connections in the system replaced by elastic spring contacts, if Assumption 10.4 holds.

10.3.4 Learning Critical Obstacle Points via Input Inference

The leader infers the reactive feedback of the follower in $v_{t+\delta}$ in (10.7) at timestep $t+2\delta$. Using this, the leader's estimate of the critical obstacle point seen by the follower at timestep $t+\delta$ is denoted as $\hat{\mathcal{C}}_{\text{cr},t+\delta}^{(l)}$. For obtaining this estimate we first need the following assumption, along with Assumptions 10.1-10.3 stated in Section 10.3.1. Let $\hat{S}_t^{(l)} \in \mathcal{Y}_l$, $\forall t \geq 0$.

Assumption 10.5 *We assume that the map from the set $\mathcal{V}$ to the set $\mathcal{Y}_l$ is invertible and $f_2(\cdot, d_{\text{cr}}, \cdot, \cdot)$ is an invertible function for any chosen value of the critical distance d_{cr}.*

We choose function $f_2(\cdot, d_{\text{cr}}, \cdot, \cdot)$ satisfying Assumption 10.5. Assumption 10.5 ensures the leader is able to uniquely infer the critical obstacle points using its estimated follower's

inputs. Satisfying Assumptions 10.1-10.3 and Assumption 10.5, the construction of $\hat{\mathcal{C}}^{(l)}_{\mathrm{cr},t+\delta}$ for model (10.3) and follower policy (10.8) is shown in detail in the Appendix. For this estimation the leader uses (10.4a) and computes estimates of the corresponding follower states as:

$$\begin{aligned} \hat{X}^{(l)}_{f,t+\delta} &= X_{l,t+\delta} - (l_l + l_f)\cos\theta_{t+\delta}, \\ \hat{Y}^{(l)}_{f,t+\delta} &= Y_{l,t+\delta} - (l_l + l_f)\sin\theta_{t+\delta}, \end{aligned} \tag{10.14}$$

and then obtains:

$$\hat{\mathcal{C}}^{(l)}_{\mathrm{cr},t+\delta} = \begin{bmatrix} \hat{X}^{(l)}_{f,t+\delta} + \hat{d}_{t+\delta}\cos(\theta_{t+\delta} - \hat{\phi}_{t+\delta}) \\ \hat{Y}^{(l)}_{f,t+\delta} + \hat{d}_{t+\delta}\sin(\theta_{t+\delta} - \hat{\phi}_{t+\delta}) \end{bmatrix}, \tag{10.15}$$

where $\hat{d}_{t+\delta}$ and $\hat{\phi}_{t+\delta}$ are the leader's estimate of $d_{t+\delta}$ and $\phi_{t+\delta}$, respectively. With the inferred $\hat{\mathcal{C}}^{(l)}_{\mathrm{cr},t}$, the leader then updates and uses:

$$\mathcal{C}_{l,t+T_s} = \mathcal{C}_{l,t} \cup \delta\mathcal{C}_{l,t+T_s} \cup \hat{\mathcal{C}}^{(l)}_{\mathrm{cr},t+\delta}, \tag{10.16}$$

where $\delta\mathcal{C}_{l,t+T_s}$ denotes the new obstacle constraints detected by the leader at the next timestep. The process is then repeated from timestep $(t+T_s)$ onward.

10.3.5 Leader-Follower Role Switching

Although the leader learns $\hat{\mathcal{C}}^{(l)}_{\mathrm{cr},t+\delta}$ and updates its controller, this can still lead to failure in avoiding obstacles in tight environments. For example, if the follower approaches a tight corner with multiple obstacles, the leader may not have sufficient time to generate a feasible trajectory for the joint system, as it does not directly detect the whole obstacle map from the follower and infers *only* the critical obstacle points detected by the follower. Therefore, switching the roles of the leader and the follower in these scenarios can be useful, enabling the leader to directly see all the obstacles in the tight corner. Such a role switching strategy of the leader and the follower is motivated by [125], where the roles are switched with a fixed frequency. In general, we define a time dependent role switching function for an agent as:

$$f_{\mathrm{swt}} : (x, \mathcal{C}, t) \mapsto \{0, 1\}, \tag{10.17}$$

where x and $\mathcal{C}$ denote the switching deciding states and obstacles of the agent, respectively, and 0 denotes no switching and 1 denotes a switch trigger.

10.3.6 The complete Algorithm

We summarize our proposed collaborative obstacle avoidance with environment learning algorithm with system model (10.3) and follower policy parametrization (10.8) in Algorithm 7. As a specific choice for (10.17), we pick:

Algorithm 7 Collaborative Obstacle Avoidance with Environment Learning

Initialize: $t=0$, $v_{(t-T_s)+\delta}, S_0$
Inputs: S_{tar}, Q_i, Q_s, d_{cr}, $\mathcal{U}$, $K_1, K_2, T, \delta, N, T_s, \mathcal{X}$
Data: $\mathcal{C}_{l,t}$, $\mathcal{C}_{f,t}$

while $t \leq T$ **do**
  Leader at t: Get u_t from (10.9)-(10.11). Apply to (10.3);
  Follower at t: Apply $v_{(t-T_s)+\delta}$ to (10.3) in $[t, t+\delta)$;
  Follower at $t+\delta$: Compute $\mathcal{C}_{\text{cr},t+\delta}$ with (10.6);
    Obtain $\hat{R}^{(f)}_{t+\delta}$ from (10.4b);
    Infer $\hat{u}_t$ (see Appendix);
    Apply $v_{t+\delta}$ in (10.13) to (10.3);
  Leader at $t+2\delta$: Obtain $\hat{R}^{(l)}_{t+\delta}$ from (10.14);
    Obtain $\hat{S}^{(l)}_{t+2\delta}$ from (10.4a);
    Estimate $\hat{v}_{t+\delta}$. Get $\hat{d}_{t+\delta}$, $\hat{\phi}_{t+\delta}$ from (10.13);
    Estimate $\hat{\mathcal{C}}^{(l)}_{\text{cr},t+\delta}$ with (10.15);
  Follower at $t+T_s$: Check $f_{\text{swt}}(\hat{R}^{(f)}_{t+\delta}, \mathcal{C}_{\text{cr},t+\delta})$ and pick switch;
  Leader at $t+T_s$: Check $f_{\text{swt}}(\hat{R}^{(l)}_{t+\delta}, \hat{\mathcal{C}}^{(l)}_{\text{cr},t+\delta})$ and pick switch;
  $t = t+T_s$;
end while

$$f_{\text{swt}}(R_{t+\delta}, \mathcal{C}_{t+\delta}) = \begin{cases} 1, \text{ if } \|R_{t+\delta} - \mathcal{C}_{t+\delta}\| \leq d_{\text{thr}}, \\ 0 \text{ otherwise,} \end{cases} \tag{10.18}$$

where d_{thr} is a chosen distance threshold value, and the follower and the leader use $\mathcal{C}_{t+\delta} = \mathcal{C}_{\text{cr},t+\delta}$, $\mathcal{C}_{t+\delta} = \hat{\mathcal{C}}^{(l)}_{\text{cr},t+\delta}$, and $R_{t+\delta} = \hat{R}^{(f)}_{t+\delta}$ and $R_{t+\delta} = \hat{R}^{(l)}_{t+\delta}$ obtained from (10.4), respectively. Having evaluated (10.18) at timestep $t+\delta$, the agents decide the role switch trigger accordingly for control design at $t+T_s$. Since the error between $\hat{R}^{(f)}_{t+\delta}$ and $\hat{R}^{(l)}_{t+\delta}$ is zero (see (10.4)), the switch happens simultaneously at $t+T_s$ without any explicit communication, if the leader has an accurate estimate (10.15). We alternately keep changing the cost in (10.9)-(10.10) with role switches, penalizing the deviation of the initial leader from S_{tar}.

Remark 10.3 (Unsynchronized Clocks) *If the leader's and the follower's clocks are unsynchronized, the order of operations shown in Algorithm 7 can no longer be ensured. However, for a small sampling time T_s, the performance of Algorithm 7 does not change noticeably. This was observed during the numerical experiments in Section 10.4 with the chosen value of T_s in Table 10.1. In such cases, check (10.18) may result in two leaders for a fraction of the sampling time, when both robots apply MPC controllers by solving (10.9)-(10.11).*

Remark 10.4 (Imperfect Estimation) *If the estimate $\hat{\mathcal{C}}^{(l)}_{\text{cr},t+\delta}$ has large errors, one may alternatively decide role switches with a fixed time period, similar to [125]. In such a case,*

the leader may not include $\mathcal{C}^{(l)}_{\mathrm{cr},t+\delta}$ *in* (10.16)*, and instead include a time-varying penalty in* (10.9) *based on its inferred obstacles.*

10.4 Numerical Experiments

We present our numerical simulations in this section. First we detail the problem setup, and then we compare the results from Algorithm 7 with two alternative strategies. We consider synchronous clocks for these experiments. The source code is available at in: *github.com/monimoyb/LeadFollowRobots*. We use Python 3.7.3 and the SLSQP solver in SciPy 1.6.

10.4.1 Experimental Setup

The parameters of the problem are shown in Table 10.1. We consider the set of obstacles in $\mathcal{C}_{l,t}$ and $\mathcal{C}_{f,t}$ as a collection of discrete point coordinates, since we simulate the detection of these obstacles with a lidar-like angle sweep. Both the leader and the follower record point cloud information of surrounding obstacles lying within a radius of 1.2 meters, with a resolution of 3.6 degrees. The critical distance, as defined in (10.6), is chosen as $d_{\mathrm{cr}} = 1.1$ meters. We solve the leader's MPC controller synthesis problem (see Appendix for this

Table 10.1: Table of parameter values. Note, the results presented are after relaxing $\delta \ll T_s$.

Parameter	Value	Parameter	Value
m_l	0.04 kg	m_f	0.04 kg
l_l	0.8 m	l_f	0.8 m
m_r	0.01 kg	T_s	0.03 s
d_{cr}	1.4m	K_2	0.5
$\bar{F}_a$	5 N	$\bar{F}_p$	5 N
$\bar{\tau}$	0.5 Nm	δ	0.02 s
N	3	T	2.7 s

approximation to (10.9)-(10.10)) with semi-definite weight matrices

$$Q_s = \mathrm{diag}(120, 4, 120, 4, 0, 0.01),$$
$$Q_i = \mathrm{diag}(0.05, 0.05, 0.01),$$

and with

$$S_{\mathrm{tar}} = [3, 0, 3.95, 0, 0, 0]^\top, \; S_0 = [7.5, 0, 7.2, 0, 0.1, 0]^\top.$$

10.4.2 Trajectory Comparison with Alternative Strategies

We now present the results of two alternative strategies and compare them with the ones from Algorithm 7. In all the following figures, the red dot represents the leader and the blue dot represents the follower. The red star denotes the target position of the leader.

No Environment Learning

The first strategy is an MPC based standard leader-follower obstacle avoidance strategy motivated by [120, 122, 123, 126, 124], where the leader applies the MPC controller (10.9)-(10.11), the follower applies (10.8), and the leader *does not infer* any obstacle information from the follower's inputs. So the set of obstacles used by the leader for MPC design is updated as:

$$\mathcal{C}_{l,t+T_s} = \mathcal{C}_{l,t} \cup \delta\mathcal{C}_{l,t+T_s}, \ \forall t \geq 0.$$

The trajectory of the joint system with this strategy is shown in Fig. 10.3. The red crosses

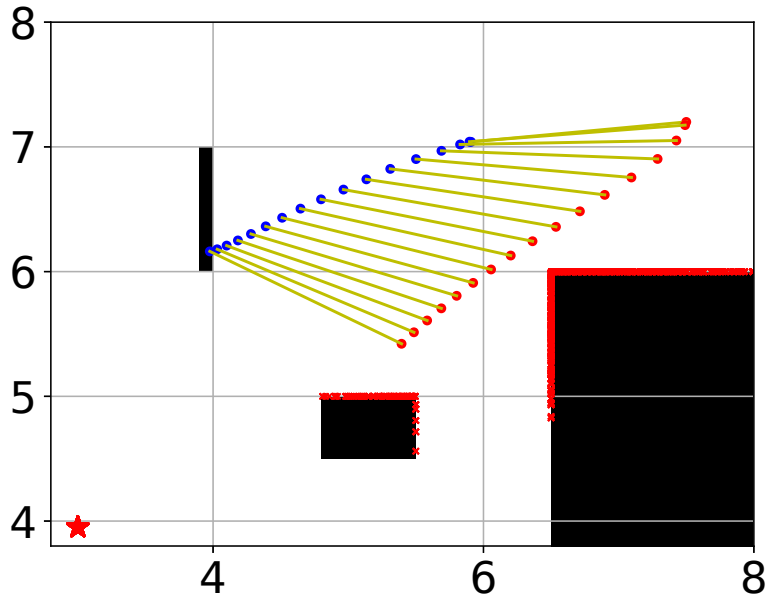


Figure 10.3: Trajectory snapshot of the rod without learning obstacles from the follower inputs. The task fails due to collision.

denote the obstacle points directly seen by the leader, which are successfully avoided. However, the rod hits the left most obstacle around position (4,6.2) on the follower's side. This obstacle remains unknown to the leader, as it is not inferred from follower inputs.

No Role Switching

The second strategy is similar to Algorithm 7, with the exception that there is *no switching* of the leader-follower roles. Such a fixed role assignment is a standard practice in the literature, e.g., [128, 129, 127, 126]. As opposed to Strategy 1, here we update the leader's known set of obstacles as (10.16), having inferred critical obstacle points (10.15) using the follower's estimated inputs. The trajectory of the joint system with this strategy is shown in Fig. 10.4. The blue crosses denote the follower's critical obstacle points inferred by the leader. As

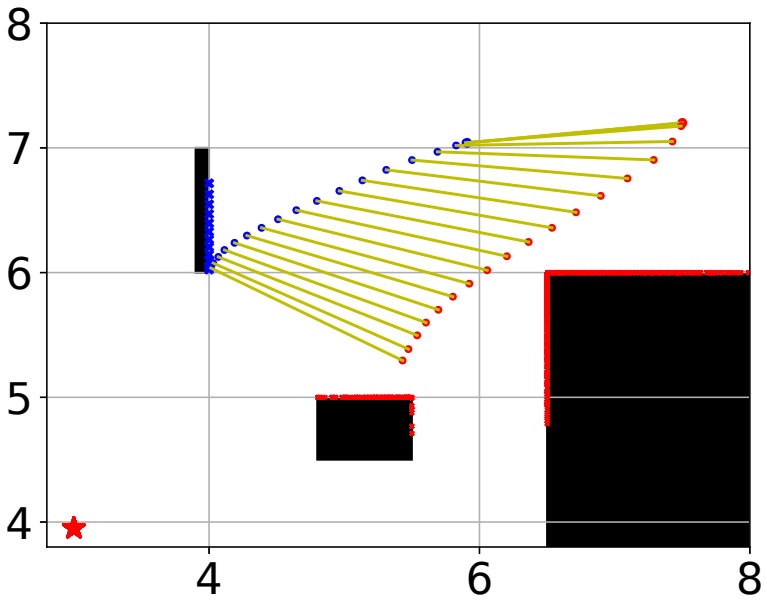


Figure 10.4: Trajectory snapshot of the rod with a fixed leader-follower role allotment and learning obstacles from the follower inputs. The task still fails due to collision.

seen in Fig. 10.4, the follower still collides with the left most obstacle around position (4,6), despite the leader learning additional blue obstacle points using the follower's feedback. This shows that a fixed role allotment here is inhibiting.

Algorithm 7

We now demonstrate the results using Algorithm 7, where we switch the roles of the leader and the follower using (10.18) and a threshold distance of $d_{\text{thr}} = 0.8$ meters. The trajectory of the joint system is shown in Fig. 10.5. As seen in Fig. 10.5, the joint system now successfully avoids all the obstacles after incorporating the leader-follower switch. After the switch, the leader directly faces the left most obstacle and collects multiple cloud points on its surface (the red crosses). These additional obstacle cloud points are missing in Fig. 10.3, where there is no obstacle inference by the leader, and also in Fig. 10.4, where the leader relies on

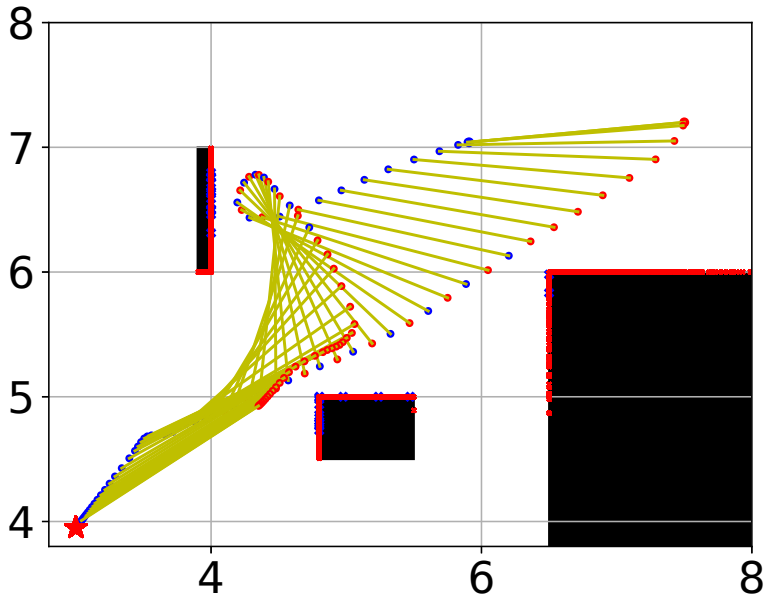


Figure 10.5: Trajectory snapshot of the rod with a switching leader-follower role allotment and learning obstacles from the follower inputs. The task succeeds.

the follower for inferring only one critical obstacle point (the blue crosses) at a time. The task also succeeds, as the initially chosen leader reaches S_{tar} by $T = 2.7$ seconds.

10.4.3 Multiple Trials with Varying Environment

We now conduct 100 trials with each of the above three strategies, with varying positions of the left most obstacle and the one at the center of the environment. The variations are contained in the purple regions shown in Fig. 10.6. The shape and the size of the obstacles are unchanged, and one of their vertices is chosen uniformly in the shown regions. Successful trials are only recorded if the joint system avoids all the obstacles and the initially chosen leader robot reaches a neighborhood of radius 0.5 meters around S_{tar} within $T = 2.7$s, i.e., 90 steps. Table 10.2 summarizes the results. Table 10.2 shows that Algorithm 7 outperforms the rest with the highest success rate, while reaching the target neighborhood fastest.

10.5 Chapter Summary

We proposed a leader-follower strategy for a two-robots collaborative transportation task in a partially known environment with obstacles. The leader solves an MPC problem at any given time with its known set of obstacles to plan a feasible trajectory and complete the task. The follower's policy is designed to assist the leader, but also react to additional obstacles in its

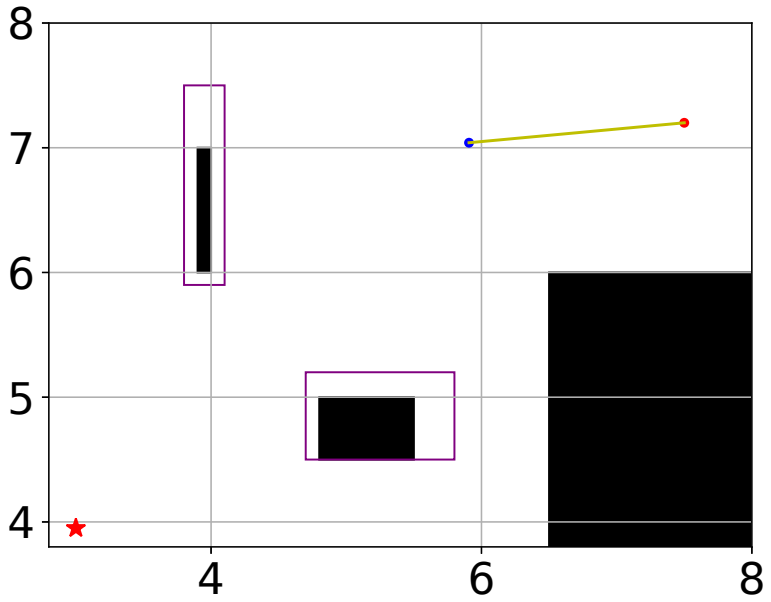


Figure 10.6: Zones containing varying obstacle positions with the given joint system's initial configuration.

Table 10.2: Strategy comparison across 100 trials. Strategy 1: No Environment Learning, Strategy 2: No Role Switching, Strategy 3: Algorithm 7. CFT denotes: Collision Free Trials.

Feature	Strategy 1	Strategy 2	Strategy 3
Successful Trials (%)	0	24	86
Collision Failures (%)	100	56	14
Timed-Out Failures (%)	0	20	0
Avg. # of Steps in a CFT	N/A	87	49

proximity which might be unseen to the leader. The difference between the predicted and the actual follower inputs is used by the leader to infer additional unseen environment constraints. This is an extension of the constraint learning concept presented in Chapter 8. We also propose a switching strategy for the leader-follower roles, improving the obstacle avoidance performance of the joint system in tight environments. In the shown numerical simulation results, our algorithm outperformed two alternative strategies, obtaining the lowest collision rate and the fastest average task completion speed.

Appendix

Tractable Approximation of (10.9)-(10.10)

We first introduce the following notation: let $\text{dist}(x, \mathcal{C})$ denote the shortest distance in ℓ_2 norm from point x to all points in the set $\mathcal{C}$. Then the approximation of (10.9)-(10.10) is:

$$\begin{aligned}
\min_{U_t} \quad & \sum_{k=1}^{N} \Bigg[(S_{t+kT_s|t} - S_{\text{tar}})^\top Q_s (S_{t+kT_s|t} - S_{\text{tar}}) + \\
& \quad + \sum_{i=1}^{d} \ell\Big(\text{dist}(\alpha_i \begin{bmatrix} s_{1,t+kT_s|t} \\ s_{3,t+kT_s|t} \end{bmatrix} + (1-\alpha_i) R_{t+kT_s|t}, \mathcal{C}_{l,t})\Big) + u_{t+(k-1)T_s|t}^\top Q_i u_{t+(k-1)T_s|t} \Bigg] \\
\text{s.t.,} \quad & S_{t+kT_s|t} = f(S_{t+(k-1)T_s|t}, u_{t+(k-1)T_s|t}, v_{t+(k-1)T_s|t}), \\
& \text{(10.10a) and (10.10c)}, \\
& \forall k \in \{1, 2, \ldots, N\},
\end{aligned} \tag{10.19}$$

where $\ell(\cdot)$ is a positive definite cost function and $\alpha_i \in [0, 1]$ for $i \in \{1, 2, \ldots, d\}$ are d sampled values of the parameter α.

Applying the Follower's Inputs for Model (10.3)

Recall the follower estimates from (10.4b). As mentioned in Section 10.3.3, the follower obtains these estimates at $t + \delta$, denoted by $\dot{\hat{X}}^{(f)}_{l,t+\delta}, \dot{\hat{Y}}^{(f)}_{l,t+\delta}$ and $\dot{\theta}_{t+\delta}$. The follower then uses the following equations:

$$\dot{\hat{X}}^{(f)}_{l,t+\delta} = \dot{\hat{X}}^{(f)}_{l,t} + \hat{q}_1 \delta, \;\; \dot{\hat{Y}}^{(f)}_{l,t+\delta} = \dot{\hat{Y}}^{(f)}_{l,t} + \hat{q}_2 \delta, \;\text{ and }\; \dot{\theta}_{t+\delta} = \dot{\theta}_t + \hat{\mathcal{T}}\delta, \;\text{ with}$$

$$\hat{\mathcal{T}} = \frac{(-F_{pf,t-T_s+\delta} l_f + \hat{F}^{(f)}_{pl,t} l_l + \tau_{f,t-T_s+\delta} + \hat{\tau}^{(f)}_{l,t})}{J},$$

$$\hat{q}_1 = -(l_l \sin\theta_t \hat{\mathcal{T}} + l_l \cos\theta_t \dot{\theta}_t^2) + \frac{1}{m}(\cos\theta_t (\hat{F}^{(f)}_{al,t} + F_{af,t-T_s+\delta}) - \sin\theta_t (\hat{F}^{(f)}_{pl,t} + F_{pf,t-T_s+\delta})),$$

$$\hat{q}_2 = (l_l \cos\theta_t \hat{\mathcal{T}} - l_l \sin\theta_t \dot{\theta}_t^2) + \frac{1}{m}(\sin\theta_t (\hat{F}^{(f)}_{al,t} + F_{af,t-T_s+\delta})) + \cos\theta_t (\hat{F}^{(f)}_{pl,t} + F_{pf,t-T_s+\delta})),$$

solves for $\hat{u}_t = [\hat{F}^{(f)}_{al,t}, \hat{F}^{(f)}_{pl,t}, \hat{\tau}^{(f)}_{l,t}]^\top$, and applies (10.13). Note, Assumption 10.4 is satisfied, as the leader's inputs appear in the above set of equations linearly and have unique solutions.

Learn Critical Obstacles with Model (10.3) and Policy (10.8)

At timestep $t+2\delta$, just after the follower applies (10.13), the leader has access to its state estimates (i.e., directly measured) using (10.4a). It then uses the following equations:

$$\dot{X}_{l,t+2\delta} = \dot{X}_{l,t+\delta} + \hat{q}_1\delta,\ \dot{Y}_{l,t+2\delta} = \dot{Y}_{l,t+\delta} + \hat{q}_2\delta,\ \text{and}\ \dot{\theta}_{t+2\delta} = \dot{\theta}_{t+\delta} + \hat{\mathcal{T}}\delta,\ \text{with}$$

$$\hat{\mathcal{T}} = \frac{(-\hat{F}^{(l)}_{pf,t+\delta} l_f + F_{pl,t} l_l + \hat{\tau}^{(l)}_{f,t+\delta} + \tau_{l,t})}{J}, \hat{\tau}^{(l)}_{f,t+\delta} = K_2\tau_{l,t},$$

$$\hat{q}_1 = -(l_l \sin\theta_{t+\delta}\hat{\mathcal{T}} + l_l\cos\theta_{t+\delta}(\dot{\theta}_{t+\delta})^2) + \frac{1}{m}(\cos\theta_{t+\delta} \times (F_{al,t} + \hat{F}^{(l)}_{af,t+\delta}) - \sin\theta_{t+\delta}(F_{pl,t} + \hat{F}^{(l)}_{pf,t+\delta})),$$

$$\hat{q}_2 = (l_l \cos\theta_{t+\delta}\hat{\mathcal{T}} - l_l\sin\theta_{t+\delta}(\dot{\theta}_{t+\delta})^2) + \frac{1}{m}(\sin\theta_{t+\delta} \times (F_{al,t} + \hat{F}^{(l)}_{af,t+\delta})) + \cos\theta_{t+\delta}(F_{pl,t} + \hat{F}^{(l)}_{pf,t+\delta})),$$

and solves for $\hat{v}_{t+\delta} = [\hat{F}^{(l)}_{af,t+\delta}, \hat{F}^{(l)}_{pf,t+\delta}, \hat{\tau}^{(l)}_{f,t+\delta}]^\top$. Note, Assumption 10.5 is satisfied, as the follower's inputs appear in the above set of equations linearly and have unique solutions. The leader infers $d_{t+\delta}$ and $\phi_{t+\delta}$ by solving:

$$\hat{\phi}_{t+\delta} = -\arctan\left(\frac{\hat{F}^{(l)}_{pf,t+\delta} - K_2 F_{pl,t}}{\hat{F}^{(l)}_{af,t+\delta} - K_2 F_{al,t}}\right), \tag{10.20a}$$

$$\|\hat{F}^{(l)}_{af,t+\delta} - K_2 F_{al,t}\|_2^2 = \left(\frac{\bar{F}_a^2(1-K_2)^2}{d_{\text{cr}}^2}\cos^2\hat{\phi}_{t+\delta} + \frac{\bar{F}_p^2(1-K_2)^2}{d_{\text{cr}}^2}\sin^2\hat{\phi}_{t+\delta}\right) \times (d_{\text{cr}} - \hat{d}_{t+\delta})^2, \tag{10.20b}$$

where $\bar{F}_{a/p}$ is the axial/perpendicular force constraint, as shown in (10.5). The leader then uses (10.20) in (10.14) to infer the critical obstacle point using (10.15).

Chapter 11

Conclusion and Future Work

In this dissertation we presented a set of algorithms for incorporating data-driven learning in robust MPC in order to lower conservatism and improve controller performance. In the following sections we present concluding remarks and lay out possible directions for future extensions of the work presented throughout the dissertation.

Concluding Summary

We primarily focused on model learning, disturbance distribution bound learning and environment constraint learning. For model learning we developed a set of adaptive MPC algorithms that adapt both parametric and non-parametric model uncertainties with data and ensure recursive satisfaction of robust constraints by the MPC. We additionally developed two novel robust MPC algorithms for LPV systems that can be utilized in these adaptive MPC approaches. For learning the disturbance distribution support we proposed a framework that can satisfy bounds on user-specified probability of constraint violations by the MPC, while avoiding conservative estimates of these support sets. We used this framework in an output feedback MPC to update the camera noise bounds and enable a robotic manipulator to play the cup-and-ball game. The manipulator's catching capabilities improved in experiments, as the camera noise support was refined with collected data. The proposed constraint learning framework in the dissertation used nonlinear classifiers to learn approximations of the environment constraints, satisfying which guarantees the satisfaction of the true constraints with a user-specified probability. We finally extended this concept of constraint learning for developing an obstacle inference and avoidance algorithm for decentralized robotic transportation tasks with only partial environment information available to each of the robots. With detailed simulations we validated that the robots inferred obstacle information via their state estimates and/or haptic feedback, incorporated this information to update their MPC planners, and adaptively switched leader-follower roles to safely complete the task.

Future Research Scope

A few very useful future extensions of the proposed work in this dissertation are as follows: (*i*) The construction of the invariant sets, Minkowski sum, Pontryagin difference, etc., required for the theoretical tools presented in Chapter 2-Chapter 9 are rendered impractical for high dimensional systems. Thus, a set of tractable tools for high dimensional systems need to be developed. (*ii*) The disturbance distribution support learning work in Chapter 7 may be extended to incorporate a wider class of distributions. Moreover, for non i.i.d. disturbances, exploration strategies can be studied and incorporated in the design for data efficient learning. (*iii*) The 2D-planar modeling of the system in Chapter 9 can be extended to 3D, which will significantly lower modeling errors currently present in the planar model. (*iv*) A more useful and generalized version of constraint learning may be attempted using expert demonstrations, as opposed to relying on potentially expensive constraint violations as done in Chapter 8. (*v*) The work in Chapter 10 may be extended to more than two robots. The assumptions of perfect state measurement are also to be relaxed. This will lead to the reformulation of the obstacle avoidance problem in terms of a *cost or reward* learning, as opposed to exact point-wise inference of obstacles.

Bibliography

[1] Benjamin Recht. "A tour of reinforcement learning: The view from continuous control". In: *Annual Review of Control, Robotics, and Autonomous Systems* 2 (2019), pp. 253–279.

[2] Marko Tanaskovic et al. "Data-driven control of nonlinear systems: An on-line direct approach". In: *Automatica* 75 (2017), pp. 1–10.

[3] Ugo Rosolia, Xiaojing Zhang, and Francesco Borrelli. "Data-driven predictive control for autonomous systems". In: *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), pp. 259–286.

[4] Manfred Morari and Jay H Lee. "Model predictive control: past, present and future". In: *Computers & Chemical Engineering* 23.4 (1999), pp. 667–682.

[5] David Q Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814.

[6] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[7] Lukas Hewing, Juraj Kabzan, and Melanie N Zeilinger. "Cautious model predictive control using gaussian process regression". In: *IEEE Transactions on Control Systems Technology* 28.6 (2019), pp. 2736–2743.

[8] Raffaele Soloperto et al. "Learning-Based Robust Model Predictive Control with State-Dependent Uncertainty". In: *IFAC Conference on Nonlinear Model Predictive Control*. Madison, Wisconsin, USA, Aug. 2018.

[9] Torsten Koller et al. "Learning-based Model Predictive Control for Safe Exploration". In: *Conference on Decision and Control*. IEEE. 2018, pp. 6059–6066.

[10] Carl Edward Rasmussen. "Gaussian processes in machine learning". In: *Summer School on Machine Learning*. Springer. 2003, pp. 63–71.

[11] Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. "Gaussian processes for data-efficient learning in robotics and control". In: *IEEE transactions on pattern analysis and machine intelligence* 37.2 (2013), pp. 408–423.

[12] Matthias Lorenzen, Frank Allgöwer, and Mark Cannon. "Adaptive Model Predictive Control with Robust Constraint Satisfaction". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 3313–3318.

[13] X. Lu and M. Cannon. "Robust Adaptive Tube Model Predictive Control". In: *American Control Conference*. IEEE. July 2019, pp. 3695–3701.

[14] Johannes Köhler et al. "Linear robust adaptive model predictive control: Computational complexity and conservatism". In: *Conference on Decision and Control*. IEEE. 2019, pp. 1383–1388.

[15] Wilbur Langson et al. "Robust model predictive control using tubes". In: *Automatica* 40.1 (2004), pp. 125–133.

[16] Diego Muñoz-Carpintero, Mark Cannon, and Basil Kouvaritakis. "Recursively feasible robust MPC for linear systems with additive and multiplicative uncertainty using optimized polytopic dynamics". In: *Conference on Decision and Control*. IEEE. 2013, pp. 1101–1106.

[17] James Fleming, Basil Kouvaritakis, and Mark Cannon. "Regions of attraction and recursive feasibility in robust MPC". In: *Mediterranean Conference on Control and Automation*. IEEE. 2013, pp. 801–806.

[18] James Fleming, Basil Kouvaritakis, and Mark Cannon. "Robust tube MPC for linear systems with multiplicative uncertainty". In: *IEEE Transactions on Automatic Control* 60.4 (2014), pp. 1087–1092.

[19] Diego Muñoz-Carpintero, Mark Cannon, and Basil Kouvaritakis. "Robust MPC strategy with optimized polytopic dynamics for linear systems with additive and multiplicative uncertainty". In: *Systems & Control Letters* 81 (2015), pp. 34–41.

[20] Anil Aswani et al. "Provably safe and robust learning-based model predictive control". In: *Automatica* 49.5 (2013), pp. 1216–1226.

[21] M. Tanaskovic et al. "Adaptive model predictive control for constrained linear systems". In: *European Control Conference (ECC)*. July 2013, pp. 382–387.

[22] Marko Tanaskovic et al. "Adaptive receding horizon control for constrained MIMO systems". In: *Automatica* 50.12 (2014), pp. 3019–3029.

[23] D Bertsekas and I Rhodes. "Recursive state estimation for a set-membership description of uncertainty". In: *IEEE Transactions on Automatic Control* 16.2 (1971), pp. 117–128.

[24] Siddharth H Nair, Monimoy Bujarbaruah, and Francesco Borrelli. "Modeling of dynamical systems via successive graph approximations". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 977–982.

[25] Tim Martin and Frank Allgöwer. "Iterative data-driven inference of nonlinearity measures via successive graph approximation". In: *Conference on Decision and Control*. IEEE. 2020, pp. 4760–4765.

[26] Shaoru Chen et al. "Robust closed-loop model predictive control via system level synthesis". In: *Conference on Decision and Control*. IEEE. 2020, pp. 2152–2159.

[27] Sarah Dean et al. "Safely learning to control the constrained linear quadratic regulator". In: *American Control Conference*. IEEE. 2019, pp. 5582–5588.

[28] M. Bujarbaruah, X. Zhang, and F. Borrelli. "Adaptive MPC with Chance Constraints for FIR Systems". In: *American Control Conference*. June 2018, pp. 2312–2317.

[29] Monimoy Bujarbaruah et al. "Adaptive MPC for Iterative Tasks". In: *Conference on Decision and Control* (2018), pp. 6322–6327.

[30] Monimoy Bujarbaruah, Siddharth H Nair, and Francesco Borrelli. "A semi-definite programming approach to robust adaptive MPC under state dependent uncertainty". In: *European Control Conference (ECC)*. IEEE. 2020, pp. 960–965.

[31] Monimoy Bujarbaruah et al. "Adaptive Stochastic MPC Under Time-Varying Uncertainty". In: *IEEE Transactions on Automatic Control* 66.6 (2020), pp. 2840–2845.

[32] Monimoy Bujarbaruah et al. "Robust MPC for Linear Systems with Parametric and Additive Uncertainty: A Novel Constraint Tightening Approach". In: *arXiv preprint arXiv:2007.00930* (2020).

[33] Monimoy Bujarbaruah et al. "A simple robust MPC for linear systems with parametric and additive uncertainty". In: *American Control Conference*. IEEE. 2021, pp. 2108–2113.

[34] X. Zhang et al. "Stochastic Model Predictive Control Using a Combination of Randomized and Robust Optimization". In: *Conference on Decision and Control*. IEEE. Florence, Italy, 2013.

[35] Roberto Tempo, Giuseppe Calafiore, and Fabrizio Dabbene. *Randomized algorithms for analysis and control of uncertain systems: with applications*. Springer Science & Business Media, 2012.

[36] Monimoy Bujarbaruah et al. "Learning robustness with bounded failure: An iterative MPC approach". In: *IFAC-PapersOnLine* 53 (2020), pp. 7085–7090.

[37] Giuseppe C Calafiore and Marco C Campi. "The scenario approach to robust control design". In: *IEEE Transactions on automatic control* 51.5 (2006), pp. 742–753.

[38] L. Armesto et al. "Efficient learning of constraints and generic null space policies". In: *International Conference on Robotics and Automation*. May 2017, pp. 1520–1526.

[39] C. Pérez-D'Arpino and J. A. Shah. "C-LEARN: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy". In: *International Conference on Robotics and Automation*. May 2017, pp. 4058–4065.

[40] Glen Chou, Dmitry Berenson, and Necmiye Ozay. "Learning constraints from demonstrations". In: *International Workshop on the Algorithmic Foundations of Robotics*. Springer. 2018, pp. 228–245.

[41] Monimoy Bujarbaruah, Charlott Vallon, and Francesco Borrelli. "Learning to satisfy unknown constraints in iterative MPC". In: *Conference on Decision and Control*. IEEE. 2020, pp. 6204–6209.

[42] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

[43] Monimoy Bujarbaruah et al. "Learning Environment Constraints in Collaborative Robotics: A Decentralized Leader-Follower Approach". In: *International Conference on Intelligent Robots and Systems*. IEEE. 2021, pp. 1636–1641.

[44] Stephen Boyd et al. *Linear matrix inequalities in system and control theory*. Vol. 15. Siam, 1994.

[45] Giuseppe C Calafiore and Laurent El Ghaoui. *Optimization models*. Cambridge university press, 2014.

[46] David Q Mayne et al. "Robust output feedback model predictive control of constrained linear systems". In: *Automatica* 42.7 (2006), pp. 1217–1222.

[47] Matthias Seeger. "Gaussian processes for machine learning". In: *International journal of neural systems* 14.02 (2004), pp. 69–106.

[48] Ignacio Alvarado et al. "Output feedback robust tube based MPC for tracking of piece-wise constant references". In: *Conference on Decision and Control*. IEEE. 2007, pp. 2175–2180.

[49] Jianchen Hu and Baocang Ding. "Output feedback robust MPC for linear systems with norm-bounded model uncertainty and disturbance". In: *Automatica* 108 (2019), p. 108489.

[50] Pierre OM Scokaert and David Q Mayne. "Min-max feedback model predictive control for constrained linear systems". In: *IEEE Transactions on Automatic control* 43.8 (1998), pp. 1136–1142.

[51] Robert W Keener. *Theoretical statistics: Topics for a core course*. Springer, 2011.

[52] Luigi Chisci, J Anthony Rossiter, and Giovanni Zappa. "Systems with persistent disturbances: predictive control with restricted constraints". In: *Automatica* 37.7 (2001), pp. 1019–1028.

[53] David Q Mayne, Maria M Seron, and SV Raković. "Robust model predictive control of constrained linear systems with bounded disturbances". In: *Automatica* 41.2 (2005), pp. 219–224.

[54] Johan Löfberg. *Minimax approaches to robust model predictive control*. Vol. 812. Linköping University Electronic Press, 2003.

[55] Paul J Goulart, Eric C Kerrigan, and Jan M Maciejowski. "Optimization over state feedback policies for robust control with constraints". In: *Automatica* 42.4 (2006), pp. 523–533.

[56] Saša V Raković et al. "Homothetic tube model predictive control". In: *Automatica* 48.8 (2012), pp. 1631–1638.

[57] Saša V Raković et al. "Fully parameterized tube MPC". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 197–202.

[58] Saša V Raković et al. "Parameterized tube model predictive control". In: *IEEE Transactions on Automatic Control* 57.11 (2012), pp. 2746–2761.

[59] Ugo Rosolia and Francesco Borrelli. "Learning Model Predictive Control for Iterative Tasks: A Computationally Efficient Approach for Linear System". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 3142–3147.

[60] Sylvia L Herbert et al. "FaSTrack: A modular framework for fast and guaranteed safe motion planning". In: *Conference on Decision and Control*. IEEE. 2017, pp. 1517–1522.

[61] Sumeet Singh et al. "Robust tracking with model mismatch for fast and safe planning: an SOS optimization approach". In: *International Workshop on the Algorithmic Foundations of Robotics*. Springer. 2018, pp. 545–564.

[62] He Yin et al. "Optimization based planner–tracker design for safety guarantees". In: *American Control Conference*. IEEE. 2020, pp. 5194–5200.

[63] Ugo Rosolia and Aaron D Ames. "Multi-rate control design leveraging control barrier functions and model predictive control policies". In: *IEEE Control Systems Letters* 5.3 (2020), pp. 1007–1012.

[64] Jerome Sieber, Samir Bennani, and Melanie N Zeilinger. "A System Level Approach to Tube-Based Model Predictive Control". In: *IEEE Control Systems Letters* 6 (2021), pp. 776–781.

[65] Aharon Ben-Tal and Arkadi Nemirovski. "Robust optimization–methodology and applications". In: *Mathematical Programming* 92.3 (2002), pp. 453–480.

[66] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004. ISBN: 0521833787.

[67] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Vol. 28. Princeton University Press, 2009.

[68] Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.

[69] Xiaojing Zhang, Monimoy Bujarbaruah, and Francesco Borrelli. "Safe and near-optimal policy learning for model predictive control using primal-dual neural networks". In: *American Control Conference*. IEEE. 2019, pp. 354–359.

[70] Xiaojing Zhang, Monimoy Bujarbaruah, and Francesco Borrelli. "Near-optimal rapid MPC using neural networks: A primal-dual policy learning framework". In: *IEEE Transactions on Control Systems Technology* 29.5 (2020), pp. 2102–2114.

[71] Mayuresh V Kothare, Venkataramanan Balakrishnan, and Manfred Morari. "Robust constrained model predictive control using linear matrix inequalities". In: *Automatica* 32.10 (1996), pp. 1361–1379.

[72] Basil Kouvaritakis, J Anthony Rossiter, and Jan Schuurmans. "Efficient robust predictive control". In: *IEEE Transactions on automatic control* 45.8 (2000), pp. 1545–1549.

[73] J Schuurmans and JA Rossiter. "Robust predictive control using tight sets of predicted states". In: *IEEE proceedings-Control theory and applications* 147.1 (2000), pp. 13–18.

[74] Mark Cannon and Basil Kouvaritakis. "Optimizing prediction dynamics for robust MPC". In: *IEEE Transactions on Automatic Control* 50.11 (2005), pp. 1892–1897.

[75] Basil Kouvaritakis and Mark Cannon. *Model predictive control: Classical, robust and stochastic*. Springer, 2016.

[76] Saša V Raković and Qifeng Cheng. "Homothetic tube MPC for constrained linear difference inclusions". In: *Chinese Control and Decision Conference*. IEEE. 2013, pp. 754–761.

[77] Ajay Gautam, Yun-Chung Chu, and Yeng Chai Soh. "Optimized dynamic policy for receding horizon control of linear time-varying systems with bounded disturbances". In: *IEEE Transactions on Automatic Control* 57.4 (2011), pp. 973–988.

[78] John C Doyle et al. "System level synthesis: A tutorial". In: *Conference on Decision and Control*. IEEE. 2017, pp. 2856–2867.

[79] James Anderson et al. "System level synthesis". In: *Annual Reviews in Control* (2019).

[80] Shaoru Chen et al. "System Level Synthesis-based Robust Model Predictive Control through Convex Inner Approximation". In: *arXiv preprint arXiv:2111.05509* (2021).

[81] Carmen Amo Alonso and Nikolai Matni. "Distributed and localized closed loop model predictive control via system level synthesis". In: *Conference on Decision and Control*. IEEE. 2020, pp. 5598–5605.

[82] Shaoru Chen et al. "Robust Model Predictive Control with Polytopic Model Uncertainty through System Level Synthesis". In: *arXiv preprint arXiv:2203.11375* (2022).

[83] James Blake Rawlings and David Q Mayne. *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[84] Sarah Dean et al. "Safely Learning to Control the Constrained Linear Quadratic Regulator". In: *arXiv preprint arXiv:1809.10121* (2018).

[85] Rene Vidal et al. "Controlled invariance of discrete time systems". In: *in Hybrid Systems: Computation and Control*. Citeseer. 1999.

[86] Ugo Rosolia, Xiaojing Zhang, and Francesco Borrelli. "Robust learning model predictive control for linear systems performing iterative tasks". In: *IEEE Transactions on Automatic Control* (2021).

[87] Yuandan Lin, Eduardo Sontag, and Yuan Wang. "Various results concerning set input-to-state stability". In: *Conference on Decision and Control*. Vol. 2. IEEE. 1995, pp. 1330–1335.

[88] Shaoru Chen. "Robust-MPC-SLS repository". In: *URL https://github.com/unstable-zeros/robust-mpc-sls* (2020).

[89] Monimoy Bujarbaruah et al. "Robust MPC for LPV systems via a novel optimization-based constraint tightening". In: *Automatica* 143 (2022). ISSN: 0005-1098.

[90] Alberto Bemporad et al. "The explicit LQR for constrained systems". In: *Automatica* 38.1 (2002), pp. 3–20.

[91] Inc Gurobi Optimization. "Gurobi optimizer reference manual". In: *URL http://www. gurobi. com* (2015).

[92] Matthias Lorenzen, Mark Cannon, and Frank Allgöwer. "Robust MPC with recursive model update". In: *Automatica* 103 (2019), pp. 461–471.

[93] Ilya Kolmanovsky and Elmer G Gilbert. "Theory and computation of disturbance invariant sets for discrete-time linear systems". In: *Mathematical Problems in Engineering* 4.4 (1998), pp. 317–367.

[94] Franco Blanchini. "Set invariance in control". In: *Automatica* 35.11 (1999), pp. 1747–1767.

[95] M. Bujarbaruah et al. "Adaptive MPC for Autonomous Lane Keeping". In: *14th International Symposium on Advanced Vehicle Control*. 2018.

[96] X. Wang, Y. Sun, and K. Deng. "Adaptive model predictive control of uncertain constrained systems". In: *American Control Conference*. June 2014, pp. 2857–2862.

[97] Marko Tanaskovic, Lorenzo Fagiano, and Vojislav Gligorovski. "Adaptive model predictive control for linear time varying MIMO systems". In: *Automatica* 105 (2019), pp. 237–245.

[98] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot. "Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments". In: *International Conference on Robotics and Automation*. May 2014, pp. 4029–4036.

[99] D. DeHaan and M. Guay. "Adaptive Robust MPC: A minimally-conservative approach". In: *2007 American Control Conference*. July 2007, pp. 3937–3942.

[100] Veronica Adetola, Darryl DeHaan, and Martin Guay. "Adaptive model predictive control for constrained nonlinear systems". In: *Systems & Control Letters* 58.5 (2009), pp. 320–326.

[101] Bernardo A Hernandez Vicente and Paul A Trodden. "Stabilizing predictive control with persistence of excitation for constrained linear systems". In: *Systems & Control Letters* 126 (2019), pp. 58–66.

[102] Huijuan Li, Anping Liu, and Linli Zhang. "Input-to-state stability of time-varying nonlinear discrete-time systems via indefinite difference Lyapunov functions". In: *ISA transactions* 77 (2018), pp. 71–76.

[103] Xiaonan Lu, Mark Cannon, and Denis Koksal-Rivet. "Robust adaptive model predictive control: Performance and parameter estimation". In: *International Journal of Robust and Nonlinear Control* 31.18 (2021), pp. 8703–8724.

[104] Ugo Rosolia and Francesco Borrelli. "Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework." In: *IEEE Transactions on Automatic Control* (2017).

[105] K. P. Wabersich and M. N. Zeilinger. "Linear Model Predictive Safety Certification for Learning-Based Control". In: *Conference on Decision and Control*. Dec. 2018, pp. 7130–7135.

[106] Antoine Girard and George J Pappas. "Approximation metrics for discrete and continuous systems". In: *IEEE Transactions on Automatic Control* 52.5 (2007), pp. 782–798.

[107] Alexander B Kurzhanski and Pravin Varaiya. "Ellipsoidal techniques for reachability analysis". In: *International Workshop on Hybrid Systems: Computation and Control*. Springer. 2000, pp. 202–214.

[108] Victor-Emmanuel Brunel et al. "Methods for estimation of convex sets". In: *Statistical Science* 33.4 (2018), pp. 615–632.

[109] Bradley Efron and Robert Tibshirani. "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy". In: *Statistical science* (1986), pp. 54–75.

[110] J. Löfberg. "YALMIP: A toolbox for modeling and optimization in MATLAB". In: *International Symposium on Computer Aided Control Systems Design*. IEEE. 2004, pp. 284–289.

[111] R. Tempo, E. W. Bai, and F. Dabbene. "Probabilistic robustness analysis: explicit bounds for the minimum number of samples". In: *Conference on Decision and Control*. Vol. 3. Dec. 1996, pp. 3424–3428.

[112] Monimoy Bujarbaruah et al. "Learning to play cup-and-ball with noisy camera observations". In: *International Conference on Automation Science and Engineering*. IEEE. 2020, pp. 372–377.

[113] Eric Hansen, Andrew Barto, and Shlomo Zilberstein. "Reinforcement learning for mixed open-loop and closed-loop control". In: *Advances in Neural Information Processing Systems* 9 (1996).

[114] C. G. Atkeson and S. Schaal. "Learning tasks from a single demonstration". In: *Proceedings of International Conference on Robotics and Automation*. Vol. 2. 1997, pp. 1706–1712.

[115] J. Z. Kolter et al. "A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving". In: *International Conference on Robotics and Automation*. 2010, pp. 839–845.

[116] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[117] Emanuel Todorov, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control". In: *International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 5026–5033.

[118] Saran Tunyasuvunakool et al. "dm_control: Software and tasks for continuous control". In: *Software Impacts* 6 (2020), p. 100022.

[119] Javier Alonso-Mora et al. "Local motion planning for collaborative multi-robot manipulation of deformable objects". In: *International Conference on Robotics and Automation*. IEEE. 2015, pp. 5495–5502.

[120] Wei Li and Rong Xiong. "Dynamical obstacle avoidance of task-constrained mobile manipulation using model predictive control". In: *IEEE Access* 7 (2019), pp. 88301–88311.

[121] Oliver Brock, Oussama Khatib, and Sriram Viji. "Task-consistent obstacle avoidance and motion behavior for mobile manipulation". In: *International Conference on Robotics and Automation*. Vol. 1. IEEE. 2002, pp. 388–393.

[122] Jaydev P Desai and Vijay Kumar. "Motion planning for cooperating mobile manipulators". In: *Journal of Robotic Systems* 16.10 (1999), pp. 557–579.

[123] Mukunda Bharatheesha et al. "Dynamic Obstacle Avoidance for Collaborative Robot Applications". In: *International Conference on Robotics and Automation*. 2017.

[124] Ola Shorinwa and Mac Schwager. "Scalable collaborative manipulation with distributed trajectory planning". In: *International Conference on Intelligent Robots and Systems*. IEEE. 2020, pp. 9108–9115.

[125] Dylan P Losey et al. "Learning from my partner's actions: Roles in decentralized robot teams". In: *Conference on Robot Learning*. PMLR. 2020, pp. 752–765.

[126] Zijian Wang and Mac Schwager. "Kinematic multi-robot manipulation with no communication using force feedback". In: *International Conference on Robotics and Automation*. IEEE. 2016, pp. 427–432.

[127] Zijian Wang and Mac Schwager. "Force-amplifying n-robot transport system for cooperative planar manipulation without communication". In: *The International Journal of Robotics Research* 35.13 (2016), pp. 1564–1586.

[128] Daniel J Stilwell and John S Bay. "Toward the development of a material transport system using swarms of ant-like robots". In: *International Conference on Robotics and Automation*. IEEE. 1993, pp. 766–771.

[129] Roderich Groß, Francesco Mondada, and Marco Dorigo. "Transport of an object by six pre-attached robots interacting via physical links". In: *International Conference on Robotics and Automation*. IEEE. 2006, pp. 1317–1323.

[130] Yasumichi Aiyama et al. "Cooperative transportation by two four-legged robots with implicit communication". In: *Robotics and Autonomous Systems* 29.1 (1999), pp. 13–19.

[131] Kazuhiro Kosuge and Tomohiro Oosumi. "Decentralized control of multiple robots handling an object". In: *International Conference on Intelligent Robots and Systems*. Vol. 1. IEEE. 1996, pp. 318–323.

[132] Hiroki Takeda et al. "Load sharing algorithm for transporting an object by two mobile robots in coordination". In: *International Conference on Intelligent Mechatronics and Automation*. 2004, pp. 374–378.

[133] Roderich Groß and Marco Dorigo. "Group transport of an object to a target that only some group members may sense". In: *International Conference on Parallel Problem Solving from Nature*. Springer. 2004, pp. 852–861.

[134] Alessandro Marino and Francesco Pierri. "A two stage approach for distributed cooperative manipulation of an unknown object without explicit communication and unknown number of robots". In: *Robotics and Autonomous Systems* 103 (2018), pp. 122–133.

[135] Alberto Bemporad and Claudio Rocchi. "Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles". In: *Conference on Decision and Control*. IEEE. 2011, pp. 7488–7493.

[136] Giuseppe Franzè, Walter Lucia, and Francesco Tedesco. "A distributed model predictive control scheme for leader–follower multi-agent systems". In: *International Journal of Control* 91.2 (2018), pp. 369–382.