Learning Model Predictive Control: Theory and Applications

by

Ugo Rosolia

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
Professor Roberto Horowitz
Associate Professor Mark Mueller
Associate Professor Benjamin Recht

Fall 2019

Learning Model Predictive Control: Theory and Applications

Abstract

Learning Model Predictive Control: Theory and Applications

by

Ugo Rosolia

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In control design, the goal is to synthesize policies which map observations to control actions. Two key elements characterize today's modern design problems: an abundance of historical data and tasks which are in full or in part repetitive. The requirements are state and input constraint satisfaction, and performance is assessed by evaluating the cost associated with the closed-loop trajectories.

In iterative control design, the policy is updated using historical data from past executions of the control task. The policy update strategy should guarantee

- (a) recursive constrain satisfaction,
- (b) iterative performance improvement with respect to previous executions, and
- (c) locally optimal behavior at convergence.

At present few methodologies are available to iteratively design predictive control policies, which satisfying the above requirements. The most common approaches resort to update the policy after performing system identification. Guarantees are provided for constraint satisfaction, but not for iterative performance improvements and/or optimality.

This thesis introduces an iterative control design methodology for constrained linear and nonlinear systems performing iterative tasks. It leads to algorithms which iteratively synthesize predictive control policies for classes of systems, where there are few, or no tools, currently available.

We will focus on three classes of discrete time dynamical systems: (i) constrained linear systems, (ii) constrained nonlinear systems, and (iii) constrained uncertain linear systems. For these three classes of systems we study iterative optimal control problems and we exploit

knowledge of the system dynamics and historical closed-loop data in the synthesis process. After each iteration of the control task, we construct a policy which uses forecast to compute safe control actions, and it is guaranteed to improve the closed-loop performance associated with stored historical data.

We call this approach the Learning Model Predictive Control (LMPC) framework. For the above systems, we introduce the policy design which exploits historical data to compute (i) a control invariant set that represents a *safe set* of states from which the control task can be completed and (ii) a control Lyapunov function which for each state of the safe set approximates the closed-loop cost of completing the task. By using the propose syntheses, we prove that properties (a),(b) and (c) can be guaranteed for the three classes of discrete time dynamical systems under consideration.

We start by presenting the LMPC design for linear and nonlinear system subject to convex and nonconvex constraints. Then, we focus on minimum time optimal control problems for linear and nonlinear systems. Afterwards, we solve the robust case for linear systems subject to bounded additive uncertainty. Finally, we present a data based policy to reduce the computational burden of the LMPC at convergence.

In the concluding part of the thesis, we present a system identification strategy tailored to iterative tasks and we demonstrate the applicability of the proposed approach through autonomous racing experiments on the Berkeley Autonomous Race Car (BARC) platform. Experimental results show that the LMPC safely learns to race a vehicle at the limits of handling.

To my parents and Agata

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to thank my advisor, Francesco Borrelli, for his support and guidance throughout my graduate studies. I am grateful for all his advice on research and professional life. I feel lucky to have had the opportunity to work in his lab. The lessons I have learned during these years have shaped me as a researcher and engineer, those will have lifelong influence on my future.

I would also like to thank Professor Roberto Horowitz, Professor Mark Mueller, Professor Benjamin Recht for serving on my dissertation committee, and Professor Recht's group for the joint lab meetings which influenced my research and broadened my horizons. I am also grateful to all the mentors that I encountered during my undergrad and master studies at Politecnico di Milano and University of Illinois at Urbana-Champaign. I would like to thank Prof. Francesco Braghin for supervising my master thesis and to Prof. Andrew Alleyne for giving me the opportunity to conduct research in his lab. This experience was of key importance for my graduate studies. Finally, I would like to thank Dr. Stijn De Bruyne for introducing me to the world of experimental testing during my internship at Siemens.

I would like to acknowledge Xiaojing (George) Zhang, which has been a great mentor. His vision, constructive feedback and suggestions were fundamentals for this thesis. I would also like to thank my colleague Jon Gonzales, without his help the experiments on the Berkeley Autonomous Race Car (BARC) would not have been possible. I would also like to acknowledge Ashwin Carvalho, Siddharth Nair and Nitin Kapania for their help testing on the full-size vehicle. Last but not least, I would like to thank Lukas Brunke, Maximilian Brunner, Martin D'Hoffschmidt, Michael Garstka, Felix Nobis, Francesco Ricciuti and Shuqi Xu. It was a pleasure and a great learning experience working with you. I hope that I made your master thesis studies interesting and rewarding.

The MPC lab has been home for the past 4 years. I would like to thank Ashwin Carvalho, Ziya Ercan, Jon Gonzales, Sarah Koehler and Theresa Lin for their warm welcome. During my stay in Berkeley I have met a long list of wonderful people Brian Cera, Fang-Chieh Chou, Roya Firoozi, Vijay Govindarajan, Yeojun Kim, Yi-Wen Liao, Greg Marcil, Siddharth Nair, Dimitris Papadimitriou, Charlott Vallon, Xu Shen, Nicola Scianca, Tony Zheng and Edward Zhu who made these years special. Thanks to Monimoy Bujarbaruah for all the fun conversations and dinner at iHouse. Thanks to all my soccer teammates and in particular to Ramon for carrying our team to the semi-final of the IM league. A special thanks goes to Jacopo Guanetti end Paolo Micalizzi for being like family away from home. Thanks Jacopo for all the support and the delicious dinners from the Artusi's recipes book, and thanks Paolo for your positive energy and enlightening conversations during all the surfing trips.

I am grateful to my wonderful family: aunts, uncles, cousins and my grandfather Attilio which is a source of inspiration. Thanks to my "little" brother Giorgio and my parents Antonio and Titti for being closed even from the other side of the ocean. Your support has been incredible during these years. Finally, I want to thank Agata for being on my side everyday since high-school and for her unconditional love. I cannot image a past and future without you by my side.

Chapter 1

Introduction

In control design, the goal is to synthesize policies which map observations to control actions. Two key elements characterize today's modern design problems: abundance of historical data and tasks which are in full or in part repetitive. The increasing sensing and data storage capabilities give us precious information which can be used for control design [1]. Experimental testing may be used to assess the closed-loop performance of the system, but the control policies deployed on experimental hardware should guarantee safety. For instance, autonomous vehicles companies could evaluate fuel efficiency by leveraging the 11 terabyte of data collected every day [2]. However, safety should be guaranteed at all times during experimental testing.

In iterative control design, the policy is updated using historical data from past executions of the control task. The policy update strategy should guarantee: state and input constraint satisfaction, iterative performance improvement and (local) optimality at convergence. At present few methodologies are available to iteratively design predictive policies, which satisfy these requirements. The most common approaches resorts to update the policy after performing system identification [3, 4, 5, 6, 7, 8, 9]. These strategies guarantee safety. However, the iterative performance improvements and/or optimality properties are not analyzed, when the controller is implemented using a receding horizon strategy.

This thesis introduces an iterative control design methodology for three classes of discrete time dynamical systems: constrained linear systems, constrained nonlinear systems, and constrained uncertain linear systems. For these three classes of systems, we study iterative optimal control problems and we exploit knowledge of the system dynamics and historical closed-loop data in the synthesis process. After each iteration of the control task, we construct a policy which uses forecast to compute safe control actions, and it is guaranteed to improve or match the closed-loop performance associated with stored historical data.

Forecasting the effects of the control actions is possible when at least one of the following components is given: (*i*) a *model* to predict the system trajectory for a given initial state and input sequence, (*ii*) a *safe set* of states from which the control task can be completed using safe policy and (*iii*) a *value function*, which for a given safe policy, maps each state of the

safe set to the closed-loop cost to complete the task. These three elements play a key role in several policy synthesis strategies such as Model Predictive Control (MPC), Approximate Dynamic Programming (ADP) and Reinforcement Learning (RL) [10, 11, 12, 13, 14].

MPC is an established control methodology which systematically uses forecast to compute control actions [10]. In MPC at each time step, a *model* is used to predict the evolution of the system over a time horizon. The sequence of control actions is chosen such that the predicted trajectory safely drives the system from the current measured state to the *safe set*, and it minimizes the predicted cost over the horizon and the future cost given by a *value function*. The MPC policy applies the first predicted input to the system, and the process is repeated at the next time step based on the new measurement, yielding to a moving or receding horizon control strategy.

In ADP and RL the value function mapping states or state-action pairs to the closed-loop cost is defined over the entire state space. This value function is used to synthesis the control policy and it affects the performance of the closed-loop system. In particular, the closed-loop performance improves as the value function better approximates the *optimal* closed-loop cost. In ADP and RL, this approximation is iteratively improved by synthesizing control policies and evaluating their closed-loop performance, either with model-based simulations or with model-free experiments. A survey on policy evaluation strategies used to construct value functions is beyond the scope of this work and we refer the reader to [13, 14] for a comprehensive review on the topic.

This thesis proposes a Learning Model Predictive Control (LMPC) framework which exploits historical data and knowledge of the system dynamics to iteratively synthesize safe control policies. We consider autonomous systems performing iterative tasks and we exploit the closed-loop data in the synthesis process. As in MPC, the proposed strategy computes the control action after forecasting the evolution of the system over a time horizon. Similarly to ADP and RL strategies, we use historical data to update the safe set and the value function used in the synthesis process. We show that the proposed strategy guarantees: (i) *safety*: state and input constraints are recursively satisfied, (ii) *performance improvement*: the closed-loop cost does not increase at each execution of the control task, (iii): *stability:* the closed-loop system asymptotically completes the task. Furthermore we show that, under mild assumptions, if the policy update process has converged (i.e. using new closed-loop data in the synthesis process does not change control policy), then the closed-loop behavior is optimal for the entire task. Finally, we present a system identification strategy tailored to iterative tasks and we demonstrate the applicability of the proposed approach through autonomous racing experiments on the Berkeley Autonomous Race Car (BARC) platform. Experimental results show that the LMPC safely learns to race a vehicle at the limits of handling. Furthermore, we show that at convergence the control policy can be approximated using a model-free strategy, which significantly reduces the computational burden, while guaranteeing safety and performance bounds.

1.1 Outline

This thesis is divided in three parts. The first part, which includes Chapters 4-5, illustrates the LMPC design for deterministic systems. The second parts describes the policy synthesis strategy for uncertain systems. In particular, in Chapters 6-8 we describe three design strategies based on different parametrizations of the feedback policy, which is used to forecast the evolution of the closed-loop system. Finally, the thirds part of the thesis illustrates the experiments performed on the Berkeley Autonomous Race Car (BARC) platform. In Chapter 9, we present the system identification strategy used to implement the LMPC, and in Chapter 10 we propose a methodology to reduce the computational complexity of the controller once the learning process has converged. The contributions of the individual chapters are highlighted below.

Chapter 2, Technical Background.
In this chapter, we introduce some basic notions from control theory. We will rely on these notions throughout the thesis.

Chapter 3, Predictive Control Policies: Synthesis and Improvement.
In this chapter, we first introduce a design methodology to synthesize predictive policies which use forecast to compute control actions. Afterwards, we survey iterative strategies to improve the synthesis process exploiting historical data. Finally, we recall some of the challenges associated with the design problem, which are illustrated on a numerical example.

Chapter 4, Learning Model Predictive Control for Deterministic Systems.
This chapter describes the Learning Model Predictive Control strategy for known deterministic systems. First we show how to use historical data to construct safe sets and approximations to the value function. Afterwards, we introduce the controller design and we illustrate its properties. Finally, we test the proposed strategy on the constrained LQR problem and on minimum time dubins vehicle problem. We show that the proposed strategy iteratively explores the state space to improve the closed-loop performance while guaranteeing safety. The python code for all examples is available online at `https://github.com/urosolia/LMPC`.

Chapter 5, Learning Model Predictive Control for Time Optimal Problems.
In time optimal control problems, the goal of the controller is to steer the system from the starting point x_S to the terminal point x_F in minimum time, while satisfying state and input constraints. In this chapter, we focus on these problems, and we show how to design a Time-Varying Learning Model Predictive Controller (LMPC) which guarantees recursive constraint satisfaction, convergence in finite time and iterative performance improvement. Compared with the previous chapter, the safe set and approximation to the value function are time varying. Furthermore, we show that these quantities can be convexified to design a relaxed LMPC, which guarantees safety and performance improvement for a class of nonlinear system and convex constraints. Finally, we illustrate the effectiveness of the proposed strategies on minimum time obstacle avoidance and racing examples. The python code for all examples is available online at `https://github.com/urosolia/LMPC`.

Chapter 6, Learning Model Predictive Control for Uncertain Systems.
In this chapter, we first illustrate the challenges associated with the computation of safe sets

from stored data of uncertain systems. Afterwards, we present an LMPC design methodology, where we used tools from set theory to construct robust safe set and approximation to the value function. We show that the proposed strategy guarantees recursive robust constraint satisfaction, iterative worst-case performance improvement and convergence to a neighborhood of the origin, regardless of the disturbance realization. Furthermore, we show that the computational burden associated with the LMPC design may be reduced using sampled closed-loop trajectories. In particular, we use roll-outs of the closed-loop system to approximate the safe set and the value function. Finally, we test the proposed strategy on a uncertain double integrator exampled and on parking problem.

Chapter 7 Robust LMPC via Feedback Policy Parametrization.
As we have discussed in the previous chapters, the LMPC computes the control action by solving a finite time optimal control problem over a moving time window. When uncertainty is acting on the system, the control problem is carried over a space of feedback policies. Such space should contain the safe policies, which may be used to complete the control task from any state into the safe set. If this condition is not verified, then the control design is challenging. In this chapter, we proposed an adaptive prediction horizon strategy which allows us to pick the space of feedback policies used by the LMPC independently form the terminal safe set. First, we illustrate how to construct robust robust sets from historical data and we characterized the associated safe control policies. Then, we propose an iterative LMPC design procedure, where data generated by a robust controller at iteration j are used to design a robust LMPC at the next $j+1$ iteration. We show that this procedure allows us to iteratively enlarge the domain of the LMPC policy and it guarantees recursive constraints satisfaction, input to state stability and performance bounds for the certainty equivalent closed-loop system. The effectiveness of the proposed control scheme is illustrated on a linear system subject to bounded additive disturbance.

Chapter 8, Certainty Equivalent Learning Model Predictive Control.
In the previous chapters we presented robust LMPC strategies based on different parametrization of the control policies. The computational burden associated with these strategies increases with the respect to the nominal case, as the controller forecasts the evolution of the system using feedback policies. In this chapter, we represent the uncertain system as the summation of a certainty equivalent and an error dynamics. Then, we propose a robust LMPC policy which exploits the certainty equivalent system to predict a nominal trajectory and the error dynamics to guarantee robust constraint satisfaction. The main advantage of proposed strategy is that the on-line computational cost does not increase with respect to the nominal case. Nevertheless, we show robust constraint satisfaction and performance improvement properties for the closed-loop system.

Chapter 9, System Identification for Autonomous Racing.
In this chapter, we illustrate the system identification strategy used to implement the Learning Model Predictive Controller (LMPC) for autonomous racing. We model the autonomous racing problem as a minimum time iterative control task, where an iteration corresponds to a lap. The system trajectory and input sequence of each lap are stored and used to systematically update the controller for the next lap. The first part of this chapter, we introduce a

local LMPC which reduces the computational burden associated with the strategy proposed in Chapter 4. Afterwards, we present a system identification strategy for the autonomous racing iterative control task. We use data from previous iterations and the vehicle's kinematic equations of motion to build an affine time-varying prediction model. The effectiveness of the proposed strategy is demonstrated by experimental results on the Berkeley Autonomous Race Car (BARC) platform.

Chapter 10, Data-Based Policy.
As we have discussed in the previous chapters, the control action given by the LMPC policy is computed solving a finite time optimal control problem over a moving time horizon. This receding horizon strategy allows the controller to deviate from the previous iterations of the control task in order to improve the closed-loop performance. In this chapter, we show how to synthesize a data-based policy, which does not explore the state space, but it is able to math the closed-loop performance of the trajectories used in the synthesis process. Therefore, this strategy may be used to reduce the computational burden of the LMPC once the learning process has converged. The proposed strategy is model-free and can be applied whenever safe input and state trajectories of a system performing an iterative task are available. These trajectories, together with a user-defined cost function, are exploited to construct a piecewise affine approximation to the value function. The approximated value function is then used to compute the control action by solving a linear program. We show that for linear system subject to convex cost and constraints, the proposed strategy guarantees closed-loop constraint satisfaction and performance bounds for the closed-loop trajectory. We evaluate the proposed strategy in simulations and experiments, the latter carried out on the Berkeley Autonomous Race Car (BARC) platform. We show that the proposed strategy is able to reduce the computation time associated with the LMPC policy by one order of magnitude while achieving the same performance.

Chapter 2

Invariant Sets and Control Lyapunov Functions

We recall some definitions from set theory [10, Chapter 10] and the definition of Input-to-State-Stability, which will be used throughout this thesis.

2.1 Invariant, Reachable and Controllable Sets for Deterministic Systems

In this section, we consider the following deterministic system

$$x_{t+1} = f(x_t, u_t) \tag{2.1}$$

where the state $x_t \in \mathbb{R}^n$ and the input $x_t \in \mathbb{R}^n$. Furthermore, we introduce the state and input constraints

$$x_t \in \mathcal{X}, u_t \in \mathcal{U},\ \forall t \geq 0. \tag{2.2}$$

For system (2.1) subject to constraints (2.2) we recall the following definitions.

Definition 1 (One-step predecessor set to the set $\mathcal{S}$) *For the system* (2.1) *, we denote the* one-step predecessor set to the set $\mathcal{S}$ *as*

$$\mathrm{Pre}(\mathcal{S}) \triangleq \{x \in \mathbb{R}^n \ :\ \exists u \in \mathcal{U} \ s.t.\ f(x,u) \in \mathcal{S}\}. \tag{2.3}$$

Definition 2 (One-step controllable set to the set $\mathcal{S}$) *For the system* (2.1)*, we denote the* one-step controllable set to the set $\mathcal{S}$ *as*

$$\mathcal{K}_1(\mathcal{S}) = \mathrm{Pre}(\mathcal{S}) \cap \mathcal{X}. \tag{2.4}$$

$\mathcal{K}_1(\mathcal{S})$ is the set of states which can be driven into the target set $\mathcal{S}$ in one time step while satisfying input and state constraints. N-step controllable sets are defined by iterating $\mathcal{K}_1(\mathcal{S})$ computations.

Definition 3 (N-step controllable set $\mathcal{K}_N(\mathcal{S})$) *For a given target set $\mathcal{S} \subseteq \mathcal{X}$, the N-step controllable set $\mathcal{K}_N(\mathcal{S})$ of the system* (2.1) *subject to the constraints* (4.2) *is defined recursively as:*

$$\mathcal{K}_j(\mathcal{S}) \triangleq \text{Pre}(\mathcal{K}_{j-1}(\mathcal{S})) \cap \mathcal{X}, \quad \mathcal{K}_0(\mathcal{S}) = \mathcal{S}, \qquad j \in \{1, \dots, N\} \tag{2.5}$$

From Definition 3, all states x_0 of the system (2.1) belonging to the N-Step Controllable Set $\mathcal{K}_N(\mathcal{S})$ can be driven, by a suitable control sequence, to the target set $\mathcal{S}$ in N steps, while satisfying input and state constraints.

Definition 4 (Maximal controllable set $\mathcal{K}_\infty(\mathcal{O})$) *For a given target set $\mathcal{O} \subseteq \mathcal{X}$, the maximal controllable set $\mathcal{K}_\infty(\mathcal{O})$ for system* (2.1) *subject to the constraints in* (4.2) *is the union of all N-step controllable sets $\mathcal{K}_N(\mathcal{O})$ contained in $\mathcal{X}$ ($N \in \mathbb{N}$).*

We will use controllable sets $\mathcal{K}_N(\mathcal{O})$ where the target $\mathcal{O}$ is a control invariant set [15]. They are special sets, since in addition to guaranteeing that from $\mathcal{K}_N(\mathcal{O})$ we reach $\mathcal{O}$ in N steps, one can ensure that once it has reached $\mathcal{O}$, the system can stay there at all future time instants. These sets are called stabilizable set.

Definition 5 (N-step (maximal) stabilizable set) *For a given control invariant set $\mathcal{O} \subseteq \mathcal{X}$, the N-step (maximal) stabilizable set of the system* (2.1) *subject to the constraints* (4.2) *is the N-step (maximal) controllable set $\mathcal{K}_N(\mathcal{O})$ ($\mathcal{K}_\infty(\mathcal{O})$).*

Since the computation of Pre-set is numerically challenging for nonlinear systems, there is extensive literature on how to obtain an approximation (often conservative) of the maximal stabilizable set [16].

Definition 6 (One-step successor set from the set $\mathcal{S}$) *For system* (2.1) *we denote the* one-step successor set from the set $\mathcal{S}$ *as*

$$\text{Succ}(\mathcal{S}) \triangleq \{x \in \mathbb{R}^n \ : \ \exists x(0) \in \ \mathcal{S}, \exists u \in \mathcal{U} \text{ s.t. } f(x(0), u) = x\}. \tag{2.6}$$

Definition 7 (Positive invariant set) *A set $\mathcal{O} \subseteq \mathcal{X}$ is said to be a positive invariant set for the system* (2.1) *and the feedback policy $\pi(x)$, if*

$$x \in \mathcal{O} \to f(x, \pi(x)) \in \mathcal{O}.$$

2.2 Invariant, Reachable and Controllable Sets for Uncertain Systems

In this section, we consider uncertain linear time invariant systems. The following definitions will be used later of to synthesize robust controllers.

Definition 8 (Robust positive invariant set) *A set $\mathcal{O} \subseteq \mathcal{X}$ is said to be a robust positive invariant set for the uncertain autonomous system $x_{t+1} = Ax_t + w_t$, with $w_t \in \mathcal{W}$ if*

$$x \in \mathcal{O} \rightarrow Ax + w \in \mathcal{O}, \ \forall w \in \mathcal{W}.$$

Definition 9 (Robust control positive invariant set) *A set $\mathcal{C} \subseteq \mathcal{X}$ is said to be a robust control positive invariant set for the uncertain system $x_{t+1} = Ax_t + Bu_t + w_t$, with $w_t \in \mathcal{W}$ and $u_t \in \mathcal{U}$, if*

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

Positive robust positive invariant and robust control positive invariant will be used throughout this thesis to guarantee robust constraint satisfaction.

Definition 10 (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*

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

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

Definition 11 (N-step robust reachable set) *Given a control policy $\pi(\cdot)$ and the closed-loop system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$ with $w_t \in \mathcal{W}$ for all $t \geq 0$, we recursively define the N-step robust reachable set from the set $\mathcal{S}$ as*

$$\mathcal{R}_{t \rightarrow t+k+1}(\mathcal{S}) = \text{Succ}(\mathcal{R}_{t \rightarrow t+k}(\mathcal{S}), \mathcal{W}), \ \mathcal{R}_{t \rightarrow t}(\mathcal{S}) = \mathcal{S}$$

for $k = \{0, \ldots, N-1\}$. Robust reachable sets are also referred to as forwards reachable sets.

Given a linear time-invariant system, the N-Step robust reachable set $\mathcal{R}_{t \rightarrow t+N}(\mathcal{S}, \mathcal{W})$ collects the state which can be reached from the set $\mathcal{S}$ in N-steps.

2.3 Lyapunov and Input-to-State Stability

This section introduces the definitions of Lyapunov stability and Input-to-State Stability (ISS).

Definition 12 (Lyapunov stability) *The equilibrium point $x = 0$ of the autonomous system $x_{t+1} = f(x_k)$ is stable in the sense of Lyapunov if $\forall \epsilon > 0$ there exists $\delta > 0$ such that*

$$||x_0|| \leq \epsilon \rightarrow ||x_t|| \leq \delta, \ \forall t \geq 0.$$

In the following, we introduce sufficient conditions to verify stability of an equilibrium point of the autonomous system $x_{t+1} = f(x_k)$.

Proposition 1 *Consider the equilibrium point $x = 0$ of system $x_{t+1} = f(x_t, u_t)$. 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*

$$V(0) = 0 \text{ and } V(x) > 0, \forall x \in \Omega \setminus \{0\} \tag{2.7a}$$
$$V(f(x)) - V(x) \leq 0. \tag{2.7b}$$

Then $x = 0$ is asymptotically in the sense of Lyapunov on Ω.

Definition 13 (Lyapunov function) *A function $V(\cdot)$ satisfying conditions 2.7a-2.7b is called a Lyapunov Function.*

Next, we introduce the definition of control Lyapunov function. We will exploit control Lyapunov functions to guarantee closed-loop stability.

Definition 14 (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 invaraint set and $h(x, u)$ 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*

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

Finally, we recall the definition of Input to State Stability (ISS) of a robust invariant set [17], which extends the more widely known notion of ISS of an equilibrium point [18, 19, 20, 21]. We use the standard function classes $\mathcal{K}$, $\mathcal{K}_\infty$ and $\mathcal{KL}$ notation (see [22]) and we define the distance from a point $x \in \mathbb{R}^n$ to a set $\mathcal{O} \subseteq \mathbb{R}^n$ as

$$|x|_{\mathcal{O}} \triangleq \inf_{d \in \mathcal{O}} ||x - d||_2.$$

Definition 15 (Input to State Stability (ISS) [17]) *Let $\mathcal{O}$ be an robust positive invariant set for the autonomous system $x_{t+1} = Ax_t + B\pi(x_t) + w_t$ with $w_t \in \mathcal{W}$. We say that the closed-loop system is ISS with respect to $\mathcal{O}$ if for all $w_t \in \mathcal{W}$, $t \geq 0$, $x_0 \in \mathcal{X}$*

$$|x_{t+1}|_{\mathcal{O}} \leq \beta(|x_0|_{\mathcal{O}}, t + 1) + \gamma\big(sup_{k \in \{0,\dots,t\}} ||w_k||\big),$$

where $\beta(\cdot, \cdot)$ is a class-$\mathcal{KL}$ function and $\gamma(\cdot)$ is a class-$\mathcal{K}$ function.

Chapter 3

Predictive Control Policies: Synthesis and Improvement

In this chapter, we first introduce a design methodology to synthesize predictive policies which use forecast to compute control actions. Afterwards, we survey iterative strategies to improve the synthesis process exploiting historical data. Finally, we illustrate some of the challenges associated with the design problem using a numerical example.

3.1 Dynamic Optimization and Predictive Control

The goal of the synthesis process is to design a control policy which guarantees safety and optimal performance for the closed-loop system. These objectives can be often described by a dynamic optimization problem. For instance, consider a dynamical system described by the update equation $x_{t+1} = f(x_t, u_t)$[1], where the control input $u_t \in \mathbb{R}^d$ and the system state $x_t \in \mathbb{R}^n$. For a control task starting at $x(0)$, the control policy should apply to the system the optimal sequence of inputs $U_T^* = \{u_0^*, u_1^*, ...\}$ solving the following infinite time optimal control problem:

$$\min_{u_0, u_1, \ldots} \quad \sum_{t=0}^{\infty} h(x_t, u_t) \tag{3.1a}$$

$$\text{subject to:} \quad x_{t+1} = f(x_t, u_t), \forall t \in \{0, 1, \ldots\} \tag{3.1b}$$

$$x_t \in \mathcal{X}, u_t \in \mathcal{U}, \forall t \in \{0, 1, \ldots\} \tag{3.1c}$$

$$x_0 = x(0). \tag{3.1d}$$

In the above dynamic optimization problem, safety is encoded through the sets $\mathcal{X}$ and $\mathcal{U}$, which represent state and input constraints in (3.1c). The performance of the closed-loop system is quantified by the running cost $h(\cdot, \cdot)$ which we would like to minimize.

[1]For simplicity we assumed that the system is deterministic, the synthesis process for uncertain systems in described in Chapters 6-8.

For a particular task starting from $x(0)$, one may be tempted to compute the optimal input sequence to Problem (3.1) and apply it to the system. There are two difficulties associated with this idea. First, the infinite (or long) duration of the autonomous task can easily render the solution to (3.1) hard. Second, the model $f(\cdot)$ used to predict the evolution of the real system can be inaccurate, and even a small prediction error which cumulate at each time step can comprise the success and/or optimality of the closed-loop behavior. To alleviate both issues, it is common practice to (i) predict over an horizon N shorter than task duration and (ii) continuously measure the state of the system, say once every time step, and then recompute new control sequences with updated information from the environment. Commonly, the procedure we have described above is referred to as *Model Predictive Control (MPC)* [10, 23, 24, 25, 26, 27]. At the generic time t, an MPC policy solves the following problem

$$\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{t=0}^{N-1} h(x_t, u_t) + Q(x_N) && \text{(3.2a)}\\
\text{subject to:} \quad & x_{t+1} = f(x_t, u_t), \forall t \in \{0, \dots, N-1\} && \text{(3.2b)}\\
& x_t \in \mathcal{X}, u_t \in \mathcal{U}, \forall t \in \{0, \dots, N-1\} && \text{(3.2c)}\\
& x_N \in \mathcal{X}_N, && \text{(3.2d)}\\
& x_0 = x(t). && \text{(3.2e)}
\end{aligned}$$

where $x(t)$ is the measured state at time t.

Let $U_N^* = \{u_0^*(x(t)), \dots, u_{N-1}^*(x(t))\}$ be the optimal solution of (3.2) at time t. Then, the first element of U_N^* is applied to the system and the resulting MPC policy is:

$$\pi^{\text{MPC}}\big(x(t)\big) = u_0^*(x(t)). \tag{3.3}$$

In the above equation we use $u_i^*(x(t))$ to emphasize that the optimal solution depends on the current state $x(t)$. Later on, whenever obvious, the simpler notation u_i^* will be used. At the next time step $t+1$, the optimization problem (3.2) is solved again based on the new state $x_0 = x(t+1)$, yielding a *moving* or *receding horizon* control strategy.

Problem (3.2) compared to problem (3.1) is solved over a shorter horizon N, and it uses a terminal cost $Q(\cdot)$ and terminal constraint set $\mathcal{X}_N$ to "approximate" cost and constraints beyond the perdition horizon.The choice and the role of $Q(\cdot)$ and $\mathcal{X}_N$ are critical in MPC design and will be discussed at length later in this chapter.

We point out that it is important to distinguish between the real state $x(t)$ and input $u(t)$ of the system at time t, and the predicted states x_t and inputs u_t in the optimization problem. Indeed, often a more complex notation is used, where one differentiates between the state $x_{k|t}$ at time k predicted at time t, and the state $x_{k|t+1}$ at time k predicted at time $t+1$. We will use the complex notation later on whenever necessary.

3.2 Review of Iterative Improvement Strategies

Exploiting historical data in order to iteratively improve the performance of MPC policies has been an active theme of research in the past few decades [3, 4, 5, 6, 7, 28, 8, 29, 30, 31, 32, 33]. The key idea is to use the stored state, input and cost data to update at least one of the following elements which define the MPC policy: *i*) the prediction model in (3.2b), *ii*) a safe set used as a terminal constraint set in (3.2d) and *iii*) a value function used as a terminal cost function in (3.2a).

Policy evaluation strategies used to estimate value functions from historical data are studied in Approximate Dynamic Programming (ADP) and Reinforcement Learning (RL) [14, 13, 12]. For instance, direct strategies compute the estimate value function which best fits the closed-loop cost data over the stored states. On the other hand, in indirect strategies the estimate value function is computed by iteratively minimizing the temporal difference [34, 35]. A survey on policy evaluation strategies goes beyond the scope of this thesis, we refer the reader to [13, 14] for a comprehensive review on this topic.

The integration of MPC with system identification strategies used to estimate the prediction model has been extensively studied in the literature [3, 6, 5, 7, 4, 29, 8, 36, 37, 31, 38]. In adaptive MPC strategies [39, 40, 41, 42, 43, 44, 45, 46], set-membership approaches are used to identify the set of possible parameters and/or the domain of the uncertainty which characterize the system's model. Afterwards, robust MPC strategies for additive [47] or parametric [48, 49] uncertainty are used to guarantee robust recursive constraint satisfaction. Another strategy to identify the system dynamics is to fit a Gaussian Process (GP) to experimental data [5, 7, 4, 6, 50]. GP can be used to identify a nominal model and confidence bounds, which may be used to tighten the constraint set over the planning horizon. This strategy provides high-probability safety guarantees [50, 5, 7]. The effectiveness of GP-based strategies on experimental platform has been shown in [7, 6], where an MPC is used to race a 1/43-scale vehicle and to safely fly a drone. Regression strategies may also be used to identify a nominal model and the disturbance domain used for robust MPC design [8, 51, 52]. For instance, the authors in [8, 51, 52, 9, 53] used a linear regression strategy to identify both a nominal model and the disturbance domain used for robust MPC design. In [9, 53] the authors computed norm error bounds for the nominal model and afterward they used a system level synthesis strategy for robust control design [54].

Model-based and data-based approaches for computing safe sets have also been proposed in literature [55, 56, 57, 58, 59, 36, 37, 60, 61, 62]. In reachability-based strategies safe sets are computed solving a two players game between the controller and the disturbance [55, 56, 57]. Furthermore, these strategies provide a control policy which can be used to guarantee safety by robustly constraining the evolution of the system into the safe set [55]. Also viability theory may be used to compute safe sets [58]. The authors in [58] showed how to compute an inner approximation of the viability kernel and demonstrated the effectiveness on a RC-car set-up. The authors in [59] proposed a linear model predictive safety certification framework, where a safe sets is computed exploiting closed-loop data generated by a robust controller. In [36, 37] the authors computed safe sets combining stored trajectories with

polyhedron and ellipsoidal invariant sets. Another approach is proposed in [60] where the stored trajectories are mirrored to construct invariant sets.

3.3 Safety and Performance in Predictive Control

It is clear that the prediction model plays a crucial role in determining the success of MPC policies. If the prediction model is inaccurate, then the closed-loop system will deviate from the MPC planned trajectory. This deviation may result in poor closed-loop performance and safety constraint violation. It is less obvious that, when a perfect system model is used to design an MPC policy, the wrong choice of terminal constraint set and terminal cost function may cause undesirable closed-loop behavior.

Recall that the predictive controller (3.2) and (3.3) plans the system's trajectory over an horizon of length N, which is usually much smaller than the task duration in (3.1). With short horizon N the controller only takes shortsighted control actions, which may be unsafe or may result in poor closed-loop performance. For instance, in autonomous racing a predictive controller that plans the vehicle's trajectory over a short horizon without taking into account an upcoming curve, may accelerate to the point that safe turning would be infeasible. In this example, shortsighted control actions would force the closed-loop system to violate the safety constraints at a certain time instant. Furthermore, shortsighted control actions may lead to poor closed-loop performance. Consider an autonomous agent trying to escape from a maze along the shortest path, a predictive controller may take a sub-optimal decision, if the short prediction horizon does not allow the controller to plan a trajectory which reaches the exit.

In order to avoid such situations, we need to design controllers which take into account the evolution of the system beyond the prediction horizon. As mentioned earlier, a commonly used solution is to introduce a terminal cost $Q(\cdot)$ and terminal constraint set $\mathcal{X}_N$ in problem (3.2) in order to "approximate" cost and constraints from time N to competition of the control task. The choice and the role of $Q(\cdot)$ and $\mathcal{X}_N$ are critical in any MPC design. Properly chosen $Q(\cdot)$ and $\mathcal{X}_N$ ensure safety and performance bounds for closed-loop system despite short horizon N. In particular, the terminal constraint set $\mathcal{X}_N$ should be an *invariant set* and the terminal function $Q(\cdot)$ should be a *Lyapunov function* over $\mathcal{X}_N$, for the autonomous system controlled by $u_k = v(x_k)$[2].

3.3.1 Example: Loss of Feasibility and Optimality

In this section, we illustrate on a numerical example that shortsighted control actions may lead the controller to unsafe or poor closed-loop behaviors. We consider the following double integrator system

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t$$

[2] A formal definition of invariant set and Lyapunov function can be found in Section 2.

subject to the state constraint $x_t \in \mathcal{X} = \{x \in \mathbb{R}^2 : ||x||_\infty \leq 15\}$ and the input constraint $u_t \in \mathcal{U} = \{u \in \mathbb{R}^2 : ||u||_\infty \leq 15\}$ for all $t \geq 0$. In order to regulate the system to the origin, we implemented the MPC policy (3.3) without a terminal cost and constraint, with $Q = I$, $R = 10$ and for different horizon lengths.

Figure 3.1 shows the comparison between different closed-loop trajectories and the optimal one. We notice that for $N = 3$, the MPC is not able to regulate the system to the origin. In particular, shortsighted control actions steer the system to a state from which the finite time optimal control problem (3.2) is not feasible. On the other hand, for $N = 4$ the MPC is able to complete the regulation task, but the closed-loop behavior is suboptimal.

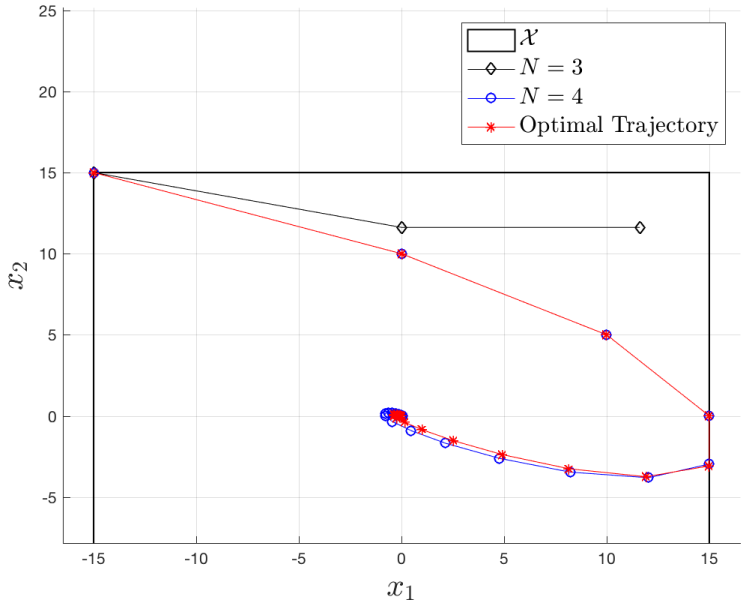


Figure 3.1: Loss of feasibility and optimality. We notice that for $N = 3$ the controller steers the system to the state $x = [11.64; 11.64]$ from which Problem (3.2) is not feasible. On the other hand, for $N = 4$ the MPC completes the regulation task, but the closed-loop trajectory is not optimal.

Chapter 4

Learning Model Predictive Control for Deterministic Systems

In this chapter, we introduce the Learning Model Predictive Controller for deterministic systems. First, we show how historical data can be used to construct control invariant sets and control Lyapunov functions. Afterwards, we exploit these quantities in the controller design, which guarantees recursive constraint satisfaction and non-decreasing closed-loop cost at each iteration. Finally, we test the controller on deterministic linear and nonlinear systems[1].

4.1 Problem Formulation

This section introduces the control problem. We consider the discrete time system

$$x_{t+1} = f(x_t, u_t), \tag{4.1}$$

where the state $x_t \in \mathbb{R}^n$ and the input $u_t \in \mathbb{R}^m$. We assume that $f(\cdot, \cdot)$ is continuous and that state and inputs are subject to the following constraints

$$x_t \in \mathcal{X},\ u_t \in \mathcal{U},\ \forall t \geq 0. \tag{4.2}$$

At the jth iteration the vectors

$$\mathbf{u}^j = [u_0^j,\ u_1^j,\ ...,\ u_t^j,\ ...], \tag{4.3a}$$
$$\mathbf{x}^j = [x_0^j,\ x_1^j,\ ...,\ x_t^j,\ ...], \tag{4.3b}$$

collect the inputs applied to system (4.1) and the corresponding state evolution. In (4.3), x_t^j and u_t^j denote the system state and the control input at time t of the jth iteration. We

[1] All code is available at `https://github.com/urosolia/LMPC`

assume that at each jth iteration the closed loop trajectories start from the same initial state, i.e.,

$$x_0^j = x_S, \ \forall j \geq 0. \tag{4.4}$$

Our objective is to design a controller which solves the following infinite horizon optimal control problem at each iteration:

$$\begin{aligned} J^*_{0\to\infty}(x_S) = \min_{u_0,u_1,\ldots} \quad & \sum_{t=0}^{\infty} h(x_t, u_t) && \text{(4.5a)} \\ \text{s.t.} \quad & x_{t+1} = f(x_t, u_t), \forall t \in \{0, 1, \ldots\} && \text{(4.5b)} \\ & x_t \in \mathcal{X}, u_t \in \mathcal{U}, \forall t \in \{0, 1, \ldots\} && \text{(4.5c)} \\ & x_0 = x_S, && \text{(4.5d)} \end{aligned}$$

where equations (4.5b) and (4.5d) represent the system dynamics and the initial condition, and (4.5c) are the state and input constraints. We assume that the stage cost $h(\cdot, \cdot)$ in equation (4.5a) is continuous and it satisfies

$$h(x_F, 0) = 0 \text{ and } h(x_t^j, u_t^j) \succ 0 \ \forall \ x_t^j \in \ \mathbb{R}^n \setminus \{x_F\}, u_t^j \in \mathbb{R}^m \setminus \{0\},$$

where the final state x_F is assumed to be a feasible equilibrium for the unforced system (4.1), that is $f(x_F, 0) = x_F$. Throughout this chapter we assume that a local optimal solution to Problem (4.5) exists and it is denoted as:

$$\begin{aligned} \mathbf{x}^* &= [x_0^*, \ x_1^*, \ ..., \ x_t^*, \ ...], \\ \mathbf{u}^* &= [u_0^*, \ u_1^*, \ ..., \ u_t^*, \ ...]. \end{aligned}$$

Remark 1 *By assumption, the stage cost $h(\cdot, \cdot)$ in (4.1) is continuous, strictly positive and zero at x_F. Thus, an optimal solution to (4.5) converges to the final point x_F, i.e.* $\lim_{t\to\infty} x_t^* = x_F$.

Remark 2 *In practical applications each iteration has a finite-time duration. It is common in the literature to adopt an infinite time formulation at each iteration for the sake of simplicity. We follow such an approach in this chapter. Our choice does not affect the practicality of the proposed method.*

Figure 4.1 shows two closed-loop trajectories (4.3) for an iterative regulation task, where the goal of the controller is to steer the system from x_S towards x_F. Next, we show that these closed-loop trajectories can be used to construct a *safe set* of states from which the control task may be completed. Afterwards, we construct a *value function* which maps each state of the safe set to the closed-loop cost.

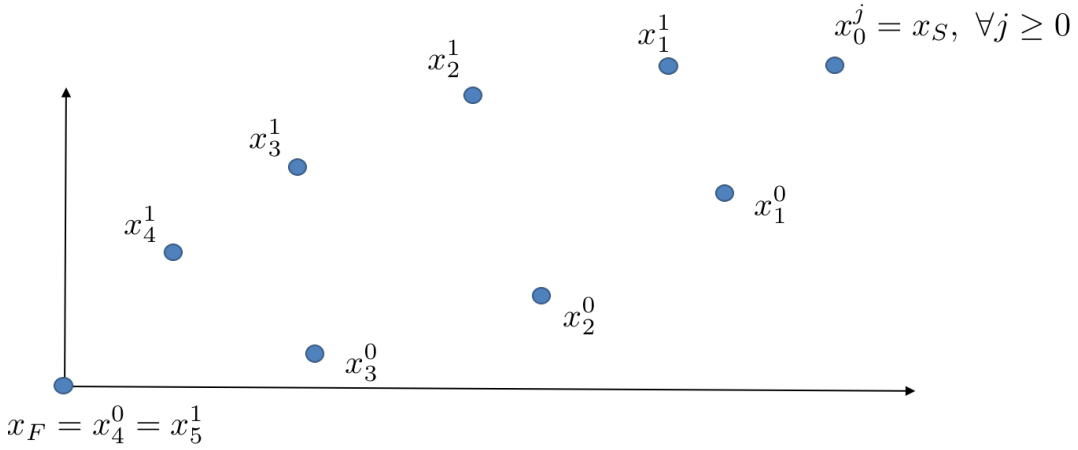


Figure 4.1: Representation of two closed-loop trajectories in the phase plane for the iterative regulation control problem. Notice that both trajectories start from the same state $x_0^0 = x_0^1 = x_s$ and reach the same terminal state $x_4^0 = x_4^1 = x_F$.

4.2 Safe Set

In this section, we exploit the iterative nature of the control problem to construct safe sets. We show that the set of states visited during a successful iteration is safe. This fact is used to define discrete, convex and local safe sets which will be used in the controller design to guarantee recursive constraint satisfaction for the closed-loop system.

4.2.1 Sampled Safe Set

We notice that, if $\pi^j(\cdot)$ is a feedback policy which is able to safely execute the desired task, then the set of states visited by closed-loop system

$$x_{t+1}^j = f(x_t^j, \pi^j(x_t^j))$$

is safe. Indeed, if at time k of the ith iteration the system's state x_k^i equals a state x_t^j which has been visited at time t of the jth iteration, then the feedback policy $\pi^j(\cdot)$ will drive the system along the jth trajectory. This obvious fact is a consequence of the system been deterministic and it implies that states visited during successful iterations are safe. More importantly, if the policy $\pi^j(\cdot)$ steers the system to an equilibrium point, then the set of visited states is a control invariant set. Thus, at iteration j we define the *sampled Safe Set* $\mathcal{SS}^j$ as the set of successful closed-loop trajectories, i.e.,

$$\mathcal{SS}^j = \left\{ \bigcup_{i \in M^j} \bigcup_{t=0}^{\infty} x_t^i \right\} \tag{4.6}$$

where the set

$$M^j = \left\{ i \in [0, j] : \lim_{t \to \infty} x_t^i = x_F \right\} \tag{4.7}$$

collects the indexes i associated with successful iterations for $i \leq j$. Notice that by definition $M^i \subseteq M^j, \forall i \leq j$ and therefore $\mathcal{SS}^i \subseteq \mathcal{SS}^j, \forall i \leq j$.

Remark 3 *Note that $\mathcal{SS}^j$ can be interpreted as a sampled subset of the maximal stabilizable set $\mathcal{K}_\infty(x_F)$ from Chapter 2 as for every point in the set, there exists a feasible control action which satisfies the state constraints and steers the system towards x_F.*

4.2.2 Convex Safe Set

In this section, we introduce the convexification of $\mathcal{SS}^j$. For linear systems and convex constraints, this convexifications will allows us to compute the control action by solving a convex optimization problem.

We define the convex safe set set as the convex hull of $\mathcal{SS}^j$ from (4.6),

$$\begin{aligned}\mathcal{CS}^j &= \text{conv}(\mathcal{SS}^j) \\ &= \{x \in \mathbb{R}^n : \forall i \in \{0, ..., j\}, \exists \boldsymbol{\lambda}^j \geq 0 \text{ such that } \sum_{i=0}^{j} \boldsymbol{\lambda}^j \mathbb{1} = 1, x = \sum_{i=0}^{j} \boldsymbol{\lambda}^j \boldsymbol{x}^j\}.\end{aligned} \tag{4.8}$$

Notice that for linear time invariant system subject convex state and input constraints, the above convex safe set is a control invariant set. Therefore, it may be used in a predictive control schema to guarantee recursive constraint satisfaction. The convex safe set and sampled convex safe set are shown in Figure 4.2.

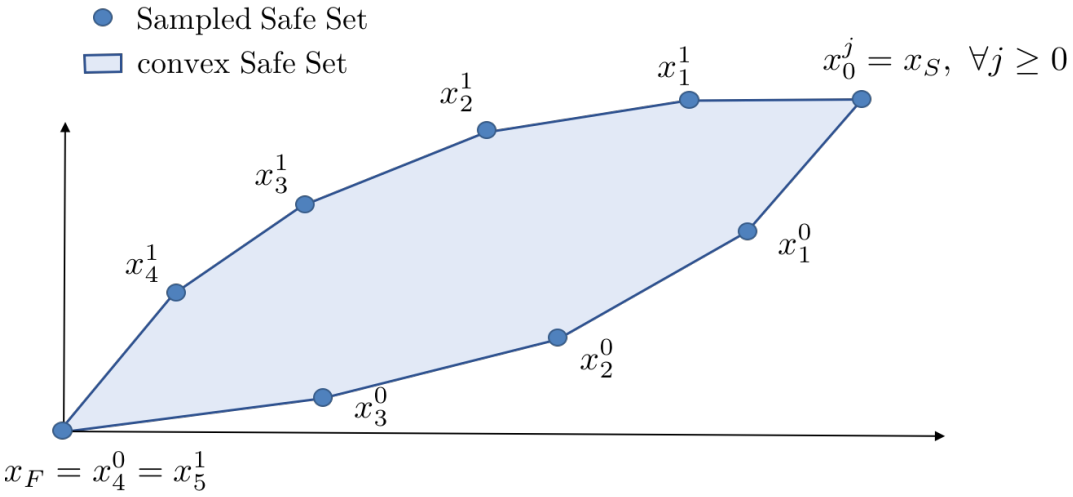


Figure 4.2: Representation of the convex safe set and sampled safe set constructed using two closed-loop trajectories.

4.2.3 Local Sampled Safe Set

The sampled safe set $\mathcal{SS}^j$ and the convex safe set $\mathcal{CS}^j$ from the previous section are constructed using the successful closed-loop trajectories up to the jth iteration. Therefore, as more iterations are performed, the number of data points used to construct these quantities increases. In this section, we introduce a local sampled safe set, which for each state $x_t^j \in \mathcal{SS}^j$ is constructed using a fixed amount of stored data. This property, as we will see later on, will allow us to reduce the computational burden associated with the controller.

We define the local safe set around $x \in \mathcal{SS}^j$ as the collection of the K-nearest neighbors to x from the lth to the jth iteration, i.e.,

$$\mathcal{LS}_l^j(x) = \bigcup_{i=l}^{j} \bigcup_{t \in \mathcal{K}^i(x)} x_t^i, \tag{4.9}$$

where, for the jth trajectory, the set $\mathcal{K}^j(x)$ collects the time indices of the K-nearest neighbors to the state x. Basically, $\mathcal{K}^j(x) = \{t_1^{j,*}, \dots, t_K^{j,*}\}$ for

$$\begin{aligned} [t_1^{j,*}, \dots, t_K^{j,*}] = \arg\min_{\mathbf{t}} \quad & \sum_{i=1}^{K} ||x_{t_i}^j - x||_2^2 \\ \text{s.t.} \quad & t_i \neq t_k, \ \forall i \neq k \\ & t_i \in \{1, 2, \dots\}, \forall i \in \{1, \dots, K\}. \end{aligned}$$

4.3 Q-functions

This section introduces sampled, convex and local Q-functions, which approximates the value function over the safe sets from the previous Section 4.2. These approximations are computed simply evaluating the cost-to-go associated with the stored closed-loop data. Later on, we will show that these Q-functions allow us to guarantee iterative performance improvement for the closed-loop system, when used as terminal cost in a predictive control schema.

4.3.1 Sampled Q-function

At time t of the jth iteration, the cost-to-go $J_{t\to\infty}^j$ associated with the closed-loop trajectory (4.3b) and input sequence (4.3a) is defined as the summation of the running cost, i.e.,

$$J_{t\to\infty}^j(x_t^j) = \sum_{k=t}^{\infty} h(x_k^j, u_k^j), \tag{4.10}$$

where $h(\cdot, \cdot)$ is the stage cost of the problem (4.5). In the above equation (4.10), x_k^j and u_k^j are the realized state and input at the jth iteration, as defined in (4.3). The above cost-to-go

is used to the define the Q-function $Q^j(\cdot)$ over the sample safe set $\mathcal{SS}^j$,

$$Q^j(x) = \begin{cases} \min\limits_{\substack{x_t^i \in \mathcal{SS}^j \\ x_t^i = x}} J^i_{t\to\infty}(x_t^i), & \text{if } x \in \mathcal{SS}^j \\ +\infty, & \text{if } x \notin \mathcal{SS}^j \end{cases}. \tag{4.11}$$

Remark 4 *The function $Q^j(\cdot)$ in (4.11) is a control Lyapunov function which assigns to every point in the sampled safe set $\mathcal{SS}^j$ the minimum cost-to-go along the trajectories in $\mathcal{SS}^j$ i.e.,*

$$\forall x \in \mathcal{SS}^j, Q^j(x) = J^{i^*}_{t^*\to\infty}(x) = \sum_{k=t^*}^{\infty} h(x_k^{i^*}, u_k^{i^*}), \tag{4.12}$$

where $x_{t^*}^{i^*}$ is function of x and it is the minimizer in (4.11):

$$x_{t^*}^{i^*} = \underset{\substack{x_t^i \in \mathcal{SS}^j \\ x_t^i = x}}{\operatorname{argmin}} J^i_{t\to\infty}(x_t^i), \quad \textit{for } x \in \mathcal{SS}^j. \tag{4.13}$$

4.3.2 Convex Q-function

In this section, we define the convex Q-function, which interpolates the cost-to-go over the convex safe set $\mathcal{CS}^j$. In particular, we define the convex Q-function $Q_c^j(\cdot)$ as the barycentric interpolation of $Q^j(\cdot)$ from (4.11),

$$\begin{aligned} Q_c^j(x) = \min_{\lambda_t^i \geq 0} \quad & \sum_{i=0}^{j}\sum_{t=0}^{\infty} \lambda_t^i J^i_{t\to\infty}(x_t^i) \\ \text{s.t.} \quad & \sum_{i=0}^{j}\sum_{t=0}^{\infty} \lambda_t^i x_t^i = x, \\ & \sum_{i=0}^{j}\sum_{t=0}^{\infty} \lambda_t^i = 1, \end{aligned} \tag{4.14}$$

where the cost-to-go $J^i_{t\to\infty}(\cdot)$ is defined in (4.10). Notice that that Problem (4.14) is a parametric LP and therefore $Q_c^j(\cdot)$ is a continuous piecewise affine convex function [10, 63]. In particular, convex Q-function is a piece-wise affine interpolation of the cost-to-go associated with the states stored in $\mathcal{SS}^j$, as shown in Figure 4.3. Finally, we underlined that for linear systems subject to convex state and input constraints, the convex Q-function is a control Lyapunov function and it will allow us to guarantee iterative closed-loop performance improvements.

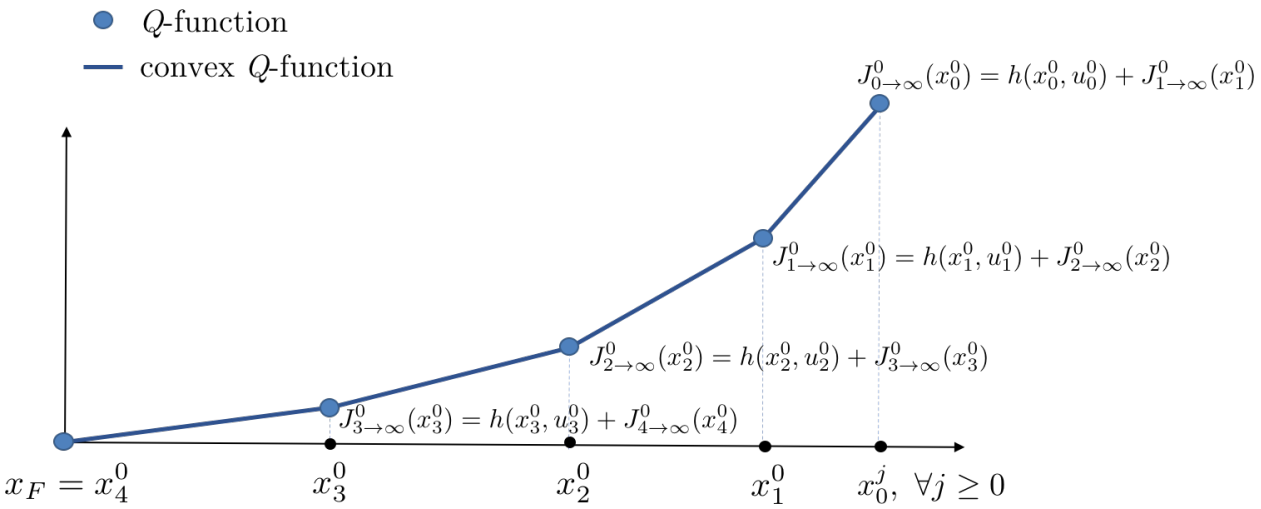


Figure 4.3: The figure shows the Q-function and the convex Q-function constructed using one stored trajectory. We notice that the Q-function is defined over a set of discrete points, and the convex Q-function is defined over a convex domain. In particular, the convex Q-function a piece-wise affine interpolation of the Q-function.

4.3.3 Local Q-function

The number of data points used to construct the Q-function and the convex Q-function from the previous sections increases as more iteration are performed. In this section, we introduce a local approximation to the value function, which for each $x^j_t \in \mathcal{SS}^j$ is constructed using a fixed number of data points. In particulate, we define the local Q-function over the local sampled safe set $\mathcal{LS}^j(\bar{x})$ around $\bar{x} \in \mathcal{SS}^j$ as the minimum cost-to-go over the stored trajectories, i.e.,

$$Q^j_l(x, \bar{x}) = \begin{cases} \min\limits_{\substack{x^i_t \in \mathcal{LS}^j(\bar{x}) \\ x^i_t = x}} J^i_{t\to\infty}(x^i_t), & \text{if } x \in \mathcal{LS}^j(\bar{x}) \\ +\infty, & \text{if } x \notin \mathcal{LS}^j(\bar{x}) \end{cases}, \tag{4.15}$$

where the cost-to-go $J^i_{t\to\infty}(\cdot)$ is defined in (4.10). The above local Q-function assigns to every state of the local sampled safe set $\mathcal{LS}^j(\bar{x})$ the minimum cost-to-go along the stored trajectories. Notice that the above local Q-function evaluated along a closed-loop trajectory is a control Lyapunov function. Later on, we will use this property to compute a state $z_t \in \mathcal{SS}^j$ which guarantees that the function $Q^j_l(\cdot, z_t)$ is a control Lyapunov function for the closed-loop system. In particular, we will compute z_t exploiting the planned trajectory at time $t-1$.

4.4 Control Design

Finally, we introduce the iterative control design procedure. At each iteration j, we store the closed-loop trajectory and we construct the safe set and Q-function, as described in Sections 4.2-4.3. Afterwards, we exploit these quantities to design the Learning Model Predictive Controller (LMPC) for the next $j+1$ iteration of the control task.

4.4.1 LMPC Policy

At time t of iteration j, the LMPC policy solves the following finite time constrained optimal control problem

$$
\begin{aligned}
J^{\text{LMPC},j}_{t\to t+N}(x^j_t) = \min_{u_{t|t},\dots,u_{t+N-1|t}} \quad & \left[\sum_{k=t}^{t+N-1} h(x_{k|t},u_{k|t}) + Q^{j-1}(x_{t+N|t})\right] && \text{(4.16a)}\\
\text{s.t.} \quad & x_{k+1|t} = f(x_{k|t},u_{k|t}), \forall k \in \{t,\dots,t+N-1\} && \text{(4.16b)}\\
& x_{k|t} \in \mathcal{X},\ u_{k|t} \in \mathcal{U}, \forall k \in \{t,\dots,t+N-1\} && \text{(4.16c)}\\
& x_{t+N|t} \in \mathcal{SS}^{j-1}, && \text{(4.16d)}\\
& x_{t|t} = x^j_t, && \text{(4.16e)}
\end{aligned}
$$

where (4.16b) and (4.16d) represent the system dynamics and initial condition, respectively. The state and input constraints are given by (4.16c). Constraint (4.16d) forces the terminal state into the set $\mathcal{SS}^{j-1}$ defined in equation (4.6).
Let

$$
\begin{aligned}
\mathbf{u}^{*,j}_{t:t+N|t} &= [u^{*,j}_{t|t},\dots,u^{*,j}_{t+N-1|t}]\\
\mathbf{x}^{*,j}_{t:t+N|t} &= [x^{*,j}_{t|t},\dots,x^{*,j}_{t+N|t}]
\end{aligned} \tag{4.17}
$$

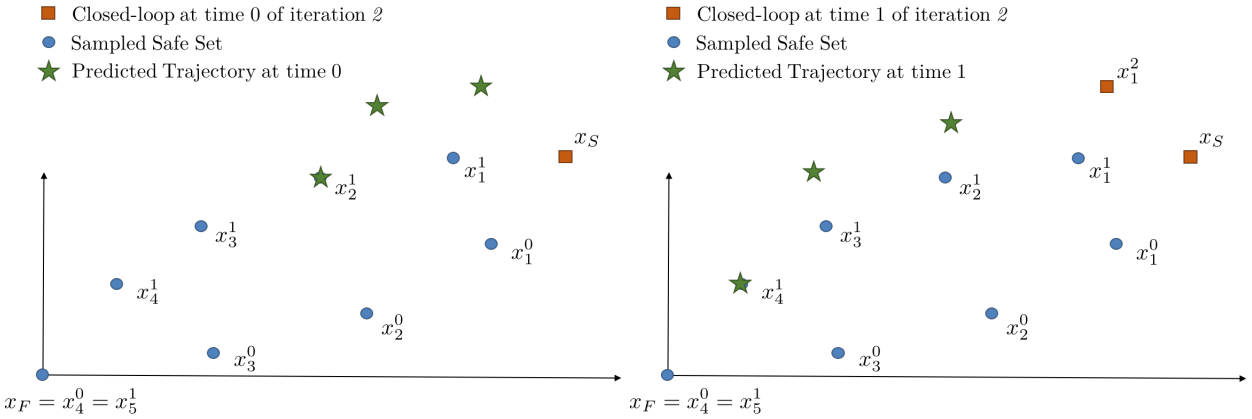
be the optimal solution of (4.16) at time t of the jth iteration and $J^{\text{LMPC},j}_{t\to t+N}(x^j_t)$ the corresponding optimal cost. Then, at time t of the iteration j, the first element of $\mathbf{u}^{*,j}_{t:t+N|t}$ is applied to the system (4.1)

$$
u^j_t = \pi^{\text{LMPC},j}(x^j_t) = u^{*,j}_{t|t}. \tag{4.18}
$$

The finite time optimal control problem (4.16) is solved at time $t+1$, based on the new state $x_{t+1|t+1} = x^j_{t+1}$, yielding a *moving* or *receding horizon* control strategy.

Assumption 1 *At iteration $j=1$ we assume that $\mathcal{SS}^{j-1} = \mathcal{SS}^0$ is a non-empty set and that the trajectory $\mathbf{x}^0 \in \mathcal{SS}^0$ is feasible and convergent to x_F.*

Assumption 1 is not restrictive in practice for a number of applications. For instance, in autonomous racing one can always run a path following controller at very low speed to obtain a feasible state and input sequence.



(a) LMPC predicted trajectory at time $t = 0$.

(b) LMPC predicted trajectory at time $t = 1$.

Figure 4.4: The blue dots represent the samples safe set from 4.6 and the green stars represent the planned open-loop trajectory given by the optimal solution to problem (4.17). Finally, the red squares represent the closed-loop trajectory. Notice that at time $t = 0$ the state of the system equals the starting states x_S, and at time $t = 1$ the state of the system equals the first predicted state at the previous time step $t = 0$.

Figures 4.4a-4.4b illustrate how the LMPC works. In particular, we reported the safe set, the planned open-loop trajectory and the closed-loop trajectory for two consecutive time instants. Notice that at time $t = 0$ of iteration $j = 2$, the predicted trajectory starts from x_S and lands on the stored state $x_2^1 \in \mathcal{SS}^1$. As shown in Figure 4.4a the predicted trajectory is not required to lay into the safe set and the controller is free to explore the state space. As a results, at time $t = 1$ the closed-loop trajectory deviates from the safe set. Figure 4.4b shows that the state of the system x_1^2 does belong to the safe set, however the controller still plans an open-loop trajectory which steers the system back to a visited state. This strategy allows the LMPC to safely explore the state space in order to iteratively improve the closed-loop performance, as we will see in Section 4.5.

4.4.2 LMPC Relaxations

The terminal constraint set in (4.16d) is the sampled safe set $\mathcal{SS}^j$. Therefore, Problem (4.16) is a Mixed Integer (MI) optimization problem and computing the control action given by the LMPC policy is expensive. In this section, we show two strategies which may be used to reduce the computational burden.

4.4.2.1 Convex LMPC

The computational cost may be reduced convexifying the terminal cost and constraint in Problem (4.16). In particular, we use the convex safe set (4.8) as terminal constraint and

the convex Q-function (4.14) as terminal cost. For linear systems subject to convex state and input constraints, we solve the following convex optimization problem

$$J^{\text{LMPC},j}_{c,t\to t+N}(x^j_t) = \min_{\substack{u^j_{t|t},\dots,u^j_{t+N-1|t}\\ \lambda^0_0,\dots,\lambda^{j-1}_0,\dots}} \quad \left[\sum_{k=t}^{t+N-1} h(x_{k|t},u_{k|t}) + \sum_{i=0}^{j-1}\sum_{k=0}^{\infty}\lambda^j_k J^i_{k\to\infty}(x^i_k)\right] \tag{4.19a}$$

$$\text{s.t.} \quad x^j_{k+1|t} = f(x_{k|t},u_{k|t}), \forall k \in \{t,\dots,t+N-1\} \tag{4.19b}$$

$$x_{k|t} \in \mathcal{X},\ u_{k|t} \in \mathcal{U}, \forall k \in \{t,\dots,t+N-1\} \tag{4.19c}$$

$$\lambda^i_k \geq 0, \forall i \in \{0,1,\dots,\}, k \in \{0,1,\dots,\} \tag{4.19d}$$

$$x_{t+N|t} = \sum_{i=0}^{j-1}\sum_{k=0}^{\infty}\lambda^i_k x^i_k,\ \sum_{i=0}^{j-1}\sum_{k=0}^{\infty}\lambda^i_k = 1, \tag{4.19e}$$

$$x_{t|t} = x^j_t. \tag{4.19f}$$

Let $[u^{j,*}_{t|t},\dots,u^{j,*}_{t+N-1|t}]$ be the optimal input sequence to the above problem. Then, the convex LMPC policy applies to the system (4.1) the first element of the optimizer vector

$$u^j_t = \pi^{\text{LMPC},j}_c(x^j_t) = u^{j,*}_{t|t}. \tag{4.20}$$

Later on we will show that for linear system, convex cost and constraints, this strategy guarantees recursive constraint satisfaction and non-increasing closed-loop cost at each iteration, as shown in the Section 4.7.4 of the Appendix.

4.4.2.2 Local LMPC

The computational burden associated with the LMPC (4.16) and (4.18) may be reduced using the local convex safe set $LS^j(\cdot)$ as terminal constraint and the local Q-function $Q^j_l(\cdot,\cdot)$ as a terminal cost function. In particular, we define the following finite time optimal control problem

$$J^{\text{LMPC},j}_{l,t\to t+N}(x^j_t,z^j_t) = \min_{u_{t|t},\dots,u_{t+N-1|t}} \quad \left[\sum_{k=t}^{t+N-1} h(x_{k|t},u_{k|t}) + Q^{j-1}_l(x_{t+N|t},z^j_t)\right] \tag{4.21a}$$

$$\text{s.t.} \quad x_{k+1|t} = f(x_{k|t},u_{k|t}), \forall k \in \{t,\dots,t+N-1\} \tag{4.21b}$$

$$x_{k|t} \in \mathcal{X},\ u_{k|t} \in \mathcal{U}, \forall k \in \{t,\dots,t+N-1\} \tag{4.21c}$$

$$x_{t+N|t} \in\ \mathcal{LS}^{j-1}(z^j_t), \tag{4.21d}$$

$$x_{t|t} = x^j_t, \tag{4.21e}$$

where, for $x^{i^*}_{t^*} = \text{argmin}_{x\in\mathcal{LS}^j(z^j_t)} Q^{j-1}_l(x,z^j_t)$, the the vector

$$z^j_{t+1} = \begin{cases} x^{j-1}_N & \text{If } t+1=0 \\ x^{i^*}_{t^*+1} & \text{Otherwise} \end{cases} \tag{4.22}$$

represents a candidate terminal state for the planned trajectory in (4.21) at time $t+1$. Let $[u^{j,*}_{t|t}, \dots, u^{j,*}_{t+N-1|t}]$ be the optimal input sequence to (4.21). Then, the local LMPC policy applies to the system (4.1) the first element of the optimizer vector

$$u^j_t = \pi_l^{\text{LMPC},j}(x^j_t) = u^{j,*}_{t|t}. \tag{4.23}$$

Notice that the local safe set is a set of discrete points and (4.21) is a MI problem. However, the number of integer variables does not grow as more iteration of the control task are performed. In the result Section 4.6.2, we will show that this strategy allows us to reduces the computation burden of the LMPC. Furthermore, it possible to show the this strategy preserves the LMPC properties for nonlinear system, non-convex cost and non-convex constraints, as shown in the Section 4.7.5 of the Appendix.

4.5 Properties

In the this section, we show that the LMPC (4.16) and (4.18) in closed-loop with system (4.1) guarantees recursively feasibility, stability and non-increasing closed-loop cost at each iteration. We focus our discuss on the LMPC from Section 4.4.1. However, similar arguments can be used to prove similar properties for the LMPC relaxations presented in Section 4.4.2. For more details we refer to Sections 4.7.4 and 4.7.5 of the Appendix.

4.5.1 Recursive feasibility and stability

As discussed in Section 4.2, for every point in the set $\mathcal{SS}^j$, there exists a feasible control sequence that can drive the system to the terminal point x_F. The properties of $\mathcal{SS}^j$ and $Q^j(\cdot)$ are used in the next proof to show recursive feasibility and asymptotic stability of the equilibrium state x_F.

Theorem 1 *Consider system (4.1) controlled by the LMPC* (4.16) *and* (4.18). *Let $\mathcal{SS}^j$ be the sampled safe set at iteration j as defined in (4.6). Let Assumption 1 hold, then the LMPC (4.16) and (4.18) is feasible for all $t \geq 0$ and iteration $j \geq 1$. Moreover, the equilibrium point x_F is asymptotically stable for the closed loop system (4.1),* (4.16) *and (4.18) at every iteration $j \geq 1$.*

Proof *The proof can be found in Section 4.7.1 of the Appendix.*

4.5.2 Performance improvement and convergence properties

In this section we show two results. First, the jth iteration cost $J^j_{0\to\infty}(\cdot)$ does not increase as j increases. Second, if the closed-loop system converges to a steady state behavior in the iteration domain, then the steady state trajectory is a local optimal solution to the infinite horizon control problem (4.5). In the following, we use the fact the Problem (4.16) is time-invariant at each iteration j and we replace $J^{\text{LMPC},j}_{t\to t+N}(\cdot)$ with $J^{\text{LMPC},j}_{0\to N}(\cdot)$.

Theorem 2 *Consider system (4.1) in closed loop with the LMPC controller (4.16) and (4.18). Let $\mathcal{SS}^j$ be the sampled safe set at the jth iteration as defined in (4.6). Let Assumption 1 hold, then the iteration cost $J^j_{0\to\infty}(\cdot) = \sum_{t=0}^{\infty} h(x^j_t, u^j_t)$ does not increase with the iteration index j.*

Proof *The proof can be found in Section 4.7.2 of the Appendix.*

Next, we assume that the LMPC (4.16) and (4.18) converges to a steady state trajectory $x^\infty_0,\ x^\infty_1, \ldots$. We try to answer the following question: "What is the link between such steady state trajectory and an optimal solution to (4.5)?". We introduce the following finite time optimal control problem closely linked to Problem (4.5),

$$\tilde{J}^*_{t\to t+T}(x_t) = \min_{u_0,\ldots,u_{T-1}} \quad \sum_{k=0}^{T-1} h(x_k, u_k) + Q^\infty(x_T) \tag{4.24a}$$

$$\text{s.t.} \quad x_{k+1} = f(x_k, u_k), \forall k \in \{0, \ldots, T-1\} \tag{4.24b}$$

$$x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \forall k \in \{0, \ldots, T-1\} \tag{4.24c}$$

$$x_T = x^\infty_{t+T}, \tag{4.24d}$$

$$x_0 = x_t. \tag{4.24e}$$

where the running cost in (4.24a), the dynamic constraint in (4.24b), the state and input constraints in (4.24c) are the same as in (4.5).

Remark 5 *Compare Problem (4.24) with Problem (4.16). Problem (4.24) uses an horizon T, possibly longer than the horizon N of Problem (4.16). Moreover, the terminal set of Problem (4.24) is a subset of the terminal set of Problem (4.16). Therefore, for $T = N$, every optimal solution to (4.16) which is feasible Problem (4.24) is also optimal.*

For the sake of simplicity we assume that Problem (4.5) is strictly convex and discuss the non-convex case in remark 6.

Assumption 2 *Problem (4.5) is strictly convex.*

Theorem 3 *Consider system (4.1) in closed loop with the LMPC controller (4.16) and (4.18) with $N > 1$. Let $\mathcal{SS}^j$ be the sampled safe set at the jth iteration as defined in (4.6). Let Assumptions 1-2 hold and assume that the LMPC controller (4.16) and (4.18) converges to the steady state input $\mathbf{u}^\infty = \lim_{j\to\infty} \mathbf{u}^j$ and the steady state trajectory $\mathbf{x}^\infty = \lim_{j\to\infty} \mathbf{x}^j$, for iteration $j \to \infty$. Denote $\text{Int}(\mathcal{S})$ as the interior of the set $\mathcal{S}$, and recall the definition of one-step predecessor $\text{Pre}(\cdot)$ and successor $\text{Succ}(\cdot)$ sets from Chapter 2. If $x^\infty_k \in \text{Int}(\text{Pre}(x^\infty_{k+1}))$ and $x^\infty_{k+1} \in \text{Int}(\text{Succ}(x^\infty_k))$ for all $k \geq 0$, then $(\mathbf{x}^\infty_{t:t+T}, \mathbf{u}^\infty_{t:t+T})$ is the optimizer of the finite horizon optimal control problem (4.24) with initial condition $x_t = x^\infty_t$ for all $t \geq 0$ and for all $T > 0$.*

Proof *The proof to the above Theorem can be found in Section 4.7.3 of the Appendix.*

Remark 6 *When problem* (4.5) *is non-convex, only local properties can be shown. In particular if one assumes that all local optimal solutions of* (4.5) *and* (4.16) *are strict, then the proof of Theorem 3 could be modified to show local optimality of* $(\mathbf{x}^{\infty}_{0:T}, \mathbf{u}^{\infty}_{0:T})$ *for the finite horizon optimal control problem* $\tilde{J}^{*}_{0\to T}(x_S)$ *for all* $T > 0$.

4.6 Examples

In this section, we test the proposed LMPC strategies on linear and nonlinear problems. First, we use the convex LMPC from Section 4.4.2.1 to solve the constrained linear quadratic regulator problem. Afterwards, we solve a minimum time nonlinear obstacle avoidance problem using the local LMPC from Section 4.4.2.2. In all example, the controller was able to iteratively improve the closed-loop performance while satisfying state and input constraints. All python code is available online[2].

4.6.1 Constrained Linear Quadratic Regulator

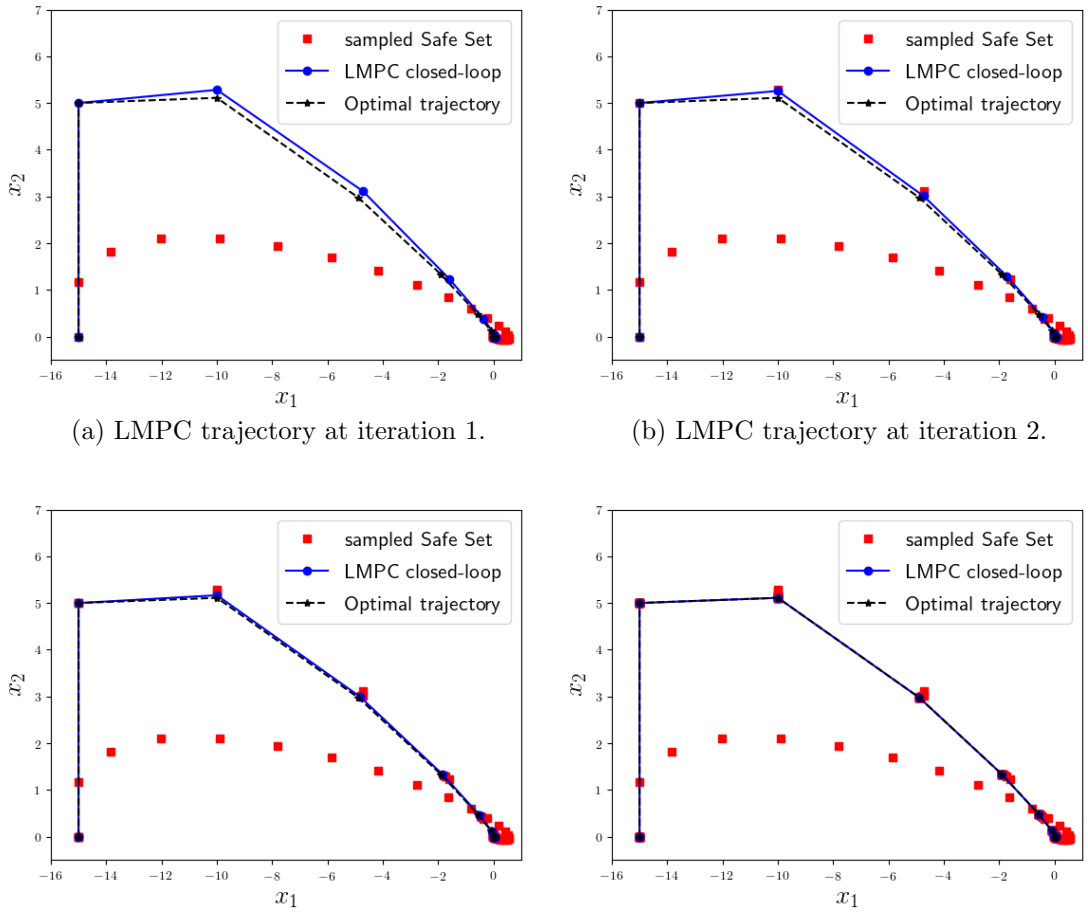
In this section, we test the proposed LMPC on the following infinite horizon constrained linear quadratic regulator (CLQR)

$$
\begin{aligned}
J^{*}_{0\to\infty}(x_S) = \min_{u_0,u_1,\dots} \quad & \sum_{t=0}^{\infty} \left[||x_t||_2^2 + ||u_t||_2^2 \right] && \text{(4.25a)}\\
\text{s.t.} \quad & x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t, \forall t \in \{0,1,\dots\} && \text{(4.25b)}\\
& \begin{bmatrix} -15 \\ -15 \end{bmatrix} \leq x_t \leq \begin{bmatrix} 15 \\ 15 \end{bmatrix}, \forall t \in \{0,1,\dots\} && \text{(4.25c)}\\
& -5 \leq u_t \leq 5, \forall t \in \{0,1,\dots\}, && \text{(4.25d)}\\
& x_0 = x_S = [-15 \;\; 1]^T. && \text{(4.25e)}
\end{aligned}
$$

Firstly, we compute a feasible solution to (4.25) using a suboptimal controller. This feasible trajectory is used to initialize the convex LMPC (4.19) and (4.20), which is implemented in CVXPY [64] with $N = 3$. At each jth iteration, the convex safe set and convex Q-function are constructed using the stored data up to iteration $j-1$. Finally, each jth iteration has an unknown fixed-time duration $\tilde{t}_j = \min\left\{t \in \mathbb{Z}_{0+} : ||x^j_t||_2^2 \leq 10^{-8}\right\}$.

Figure 4.5 shows the sampled safe set, the optimal trajectory to (4.25) and the closed-loop trajectory at iteration $j = \{1, 2, 4, 20\}$. We notice that the LMPC deviates from the

[2]All code is available at `https://github.com/urosolia/LMPC`, the first example in the folder LinearLMPC and the second one in the folder Non_LinearLMPC/DubinsObstacleAvoidance_SampledSafeSet .



(a) LMPC trajectory at iteration 1.

(b) LMPC trajectory at iteration 2.

(c) LMPC trajectory at iteration 4.

(d) LMPC trajectory at iteration 20.

Figure 4.5: Sampled safe set, optimal trajectory to (4.25) and closed-loop trajectory at iteration $j = \{1, 2, 4, 20\}$. We notice that the LMPC iteratively improves the performance of the closed-loop system, until it converges to the optimal closed-loop behavior.

first feasible trajectory used to construct the safe set in Figure 4.5a. At the first iteration, the closed-loop trajectory shown in Figure 4.5a is far from the optimal one. However, as more iteration are performed, the LMPC closed-loop trajectory gets closer to the optimal one (Fig. 4.5b and Fig. 4.5c) until the two trajectories overlap, as shown in Figure 4.5b. Finally, Figure 4.6 shows the iteration cost which is non-increasing at each iteration, until it converged to a steady-state value. We underline that the closed-loop cost converged to the optimal one within a $10^{-8}\%$ tolerance. As a result the closed-loop trajectory associates with the LMPC overlaps with the optimal solution to the CLQR (4.25), as shown in Figure 4.5d.

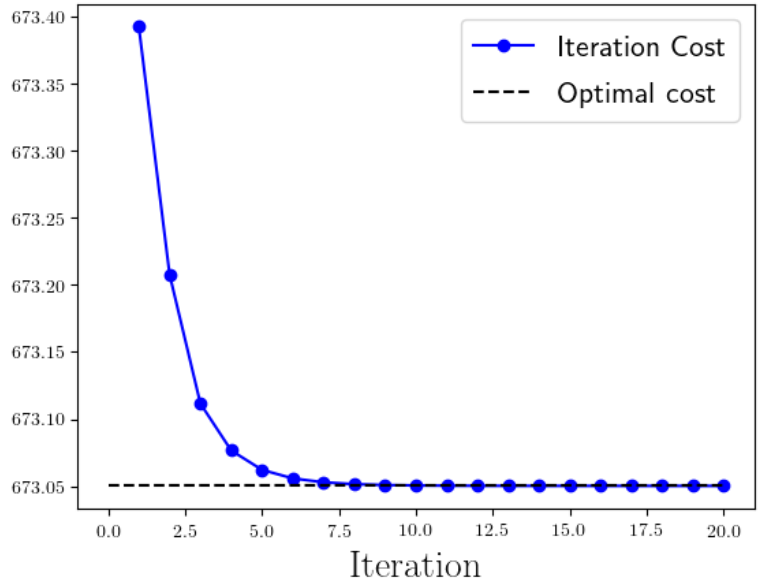


Figure 4.6: Evolution of the iteration cost through the iterations.

4.6.2 Minimum Time Dubins Car

In this section, we test the proposed LMPC on the time optimal dubins car problem [65] in discrete time. The controller's goal is to steer the system from the starting point x_S to the unforced equilibrium point x_F in minimum time, while avoiding an obstacle. The minimum time optimal control problem is formulated as the following infinite time optimal control

problem

$$J^*_{0\to\infty}(x_S) = \min_{\substack{\theta_0,\theta_1,\dots \\ a_0,a_1,\dots}} \quad \sum_{k=0}^{\infty} \mathbb{1}_{x_F}(x_t) \tag{4.26a}$$

$$\text{s.t.} \quad x_t = \begin{bmatrix} z_{t+1} \\ y_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} z_t \\ y_t \\ v_t \end{bmatrix} + \begin{bmatrix} v_t cos(\theta_t) \\ v_t sin(\theta_t) \\ a_t \end{bmatrix}, \forall t \in \{0, 1, \dots\} \tag{4.26b}$$

$$\frac{(z_t - z_{obs})^2}{a_e^2} + \frac{(y_t - y_{obs})^2}{b_e^2} \geq 1, \forall t \in \{0, 1, \dots\} \tag{4.26c}$$

$$s \leq a_t \leq s, \forall t \in \{0, 1, \dots\} \tag{4.26d}$$

$$x_0 = x_S = [0 \;\; 0 \;\; 0]^T, \tag{4.26e}$$

where the indicator function in (4.26a) is defined as

$$\mathbb{1}_{x_F}(x) = \begin{cases} 1, & \text{if } x \neq x_F \\ 0, & \text{if } x_k = x_F \end{cases}. \tag{4.27}$$

In Equation (4.26d), $s = 1$ is the known acceleration saturation limit which simulates the behavior of the friction circle [66, 67]. Equations (4.26b)-(4.26e) represent the dynamic constraint and the initial conditions, respectively. The state vector $x_t = [z_t, y_t, v_t]$ collects the car's position on the $Z - Y$ plane and the velocity, respectively. The inputs are the heading angle θ_t and the acceleration command a_t. Finally, (4.26c) represents the obstacle constraint, enforcing the system trajectory to lay outside the ellipse centered at (z_{obs},y_{obs}).

Table 4.1: Time steps to complete the task at each jth iteration

Iteration j	0	1	2	3	4	5	6	7	8	9	10
Iteration Cost	39	19	17	16	16	16	16	16	16	16	16

We implemented the local LMPC (4.21) and (4.23) with $N = 6$, the running cost $h(x_t, u_t) = \mathbb{1}_{x_F}(x_t)$ and constraints (4.26b)-(4.26c). We set $x_F = [54, 0, 0]^T$, $a_e = 8$ and $b_e = 6$. At the 0-th iteration, we computed a feasible trajectory that steers system (4.26) from x_0 to x_F using a brute force algorithm. For efficient techniques to compute collision-free trajectories in the presence of obstacle we refer to [68, 69, 70]. This feasible trajectory is used to construct the local sampled safe set $\mathcal{LS}^0(\cdot)$ and the terminal cost $Q^0_l(\cdot)$ using $K = 10$ nearest neighbors from the last 2 trajectories (i.e. $l = j - 1$ in (4.9)). Therefore, Problem (4.21) is solved computing at most 20 nonlinear optimization problems. We coded the algorithm from [62, Section 5.A.1)] and we used IPOPT [71] to solve the nonlinear programs.

We run the LMPC (4.21) and (4.18) for 10 iterations. Table 4.1 shows that the cost is decreasing until it converges to a steady state value after 4 iterations. The closed-loop

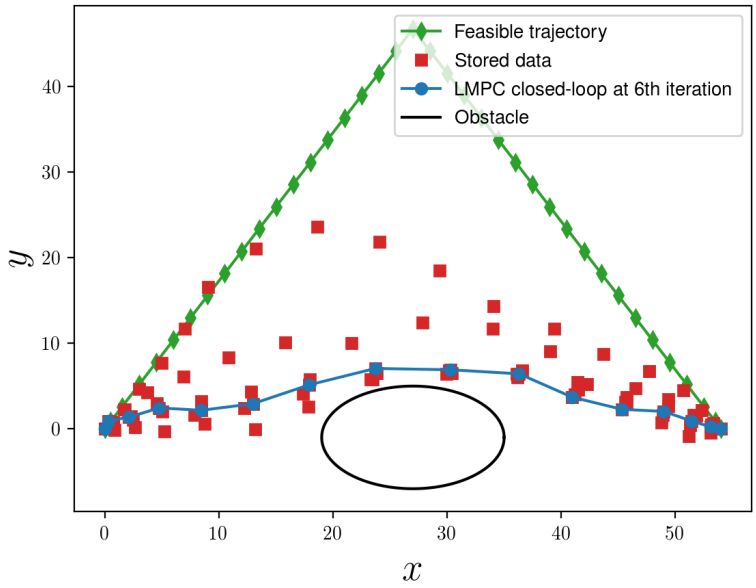


Figure 4.7: Comparison between the first feasible trajectory $\mathbf{x^0}$ and the steady state trajectory $\mathbf{x^{10}}$ at the 10th iteration.

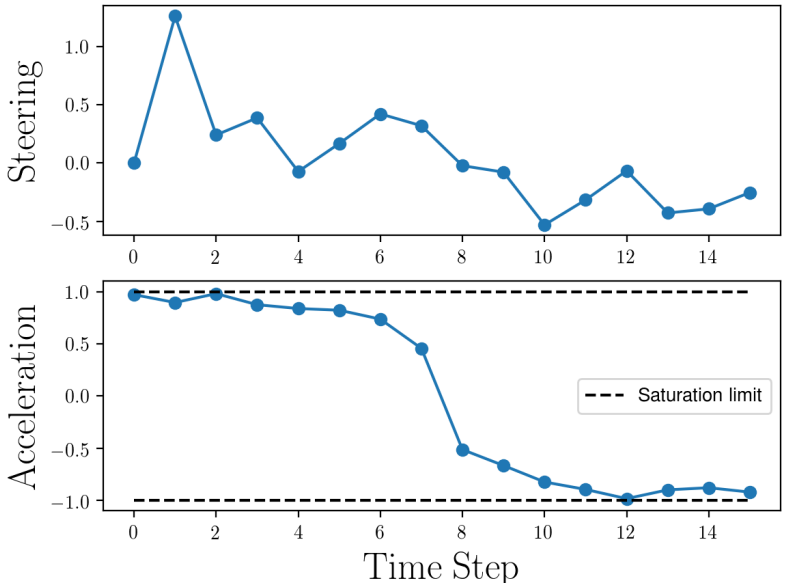


Figure 4.8: The acceleration and steering inputs associated with the closed-loop trajectory $\mathbf{x^{10}}$ at the 10th iteration.

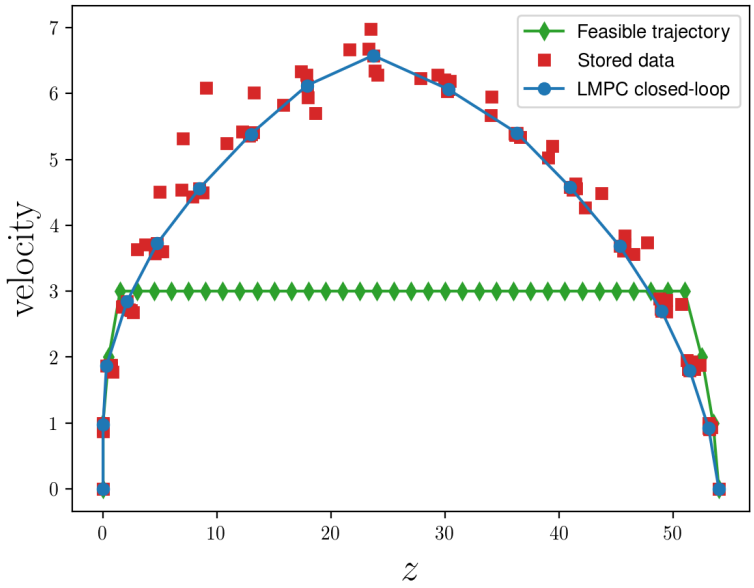


Figure 4.9: The velocity profile of the closed-loop trajectory $\mathbf{x^{10}}$ at the 10th iteration.

trajectory and the associated input sequence at the 10th iteration are reported in Figures 4.7-4.9. We notice that the acceleration input is close to saturation, as we would expect from a local optimal solution to the minimum time Problem (4.26). Similarly to a bang-bang [72] controller, the local LMPC (4.21) and (4.23) first accelerates until the dubins car reaches the midpoint between the initial and final position. Afterwards, the controller decelerates to reach the x_F with zeros velocity, as shown in Figure 4.9. Finally, we underline that in discrete time the minimum time cost is given by the number of time steps needed to reach the terminal point, therefore it is not surprising that the acceleration is not saturated all time steps. Indeed an acceleration profile similar to the one shown in Figure 4.8 that saturates the acceleration at all time steps would lead to a trajectory with the same associated cost.

4.7 Appendix

4.7.1 Proof of Theorem 1

The proof follows from standard MPC arguments.
By assumption $\mathcal{SS}^0$ is non empty. From (4.6) we have that $\mathcal{SS}^0 \subseteq \mathcal{SS}^{j-1}\ \forall j \geq 1$, and consequently $\mathcal{SS}^{j-1}$ is a non empty set. In particular, there exists a feasible trajectory $\mathbf{x}^0 \in \mathcal{SS}^0 \subseteq \mathcal{SS}^{j-1}$. From (4.4) we know that $x_0^j = x_S\ \forall j \geq 0$. At time $t = 0$ of the jth iteration the N steps trajectory

$$\mathbf{x}_{0:N}^0 = [x_0^0,\ x_1^0,\ ..., x_N^0] \in \mathcal{SS}^{j-1}, \tag{4.28}$$

and the related input sequence,

$$[u_0^0,\ u_1^0,\ ..., u_{N-1}^0], \tag{4.29}$$

satisfy input and state constrains (4.16b)-(4.16e). Therefore (4.28)-(4.29) is a feasible solution to the LMPC (4.16) and (4.18) at $t = 0$ of the jth iteration.
Assume that at time t of the jth iteration the LMPC (4.16) and (4.18) is feasible and let $\mathbf{x}_{t:t+N|t}^{*,j}$ and $\mathbf{u}_{t:t+N|t}^{*,j}$ be the optimal trajectory and input sequence, as defined in (4.17). From (4.16b) and (4.18) the realized state and input at time t of the jth iteration are given by

$$\begin{aligned} x_t^j &= x_{t|t}^{*,j}, \\ u_t^j &= u_{t|t}^{*,j}. \end{aligned} \tag{4.30}$$

The terminal constraint (4.16d) enforces $x_{t+N|t}^{*,j} \in \mathcal{SS}^{j-1}$ and, from (4.12),

$$Q^{j-1}(x_{t+N|t}^{*,j}) = J_{t^*\to\infty}^{i^*}(x_{t+N|t}^{*,j}) = \sum_{k=t^*}^{\infty} h(x_k^{i^*}, u_k^{i^*}). \tag{4.31}$$

Note that $x_{t^*+1}^{i^*} = f(x_{t^*}^{i^*}, u_{t^*}^{i^*})$ and by the definition (4.13) $x_{t^*}^{i^*} = x_{t+N|t}^{*,j}$. Since the state update in (4.1) and (4.16b) are assumed identical we have that

$$x_{t+1}^j = x_{t+1|t}^{*,j}. \tag{4.32}$$

At time $t + 1$ of the jth iteration the input sequence

$$[u_{t+1|t}^{*,j},\ u_{t+2|t}^{*,j},\ ...,\ u_{t+N-1|t}^{*,j},\ u_{t^*}^{i^*}], \tag{4.33}$$

and the related feasible state trajectory

$$[x_{t+1|t}^{*,j},\ x_{t+2|t}^{*,j},\ ...,\ x_{t+N-1|t}^{*,j},\ x_{t^*}^{i^*},\ x_{t^*+1}^{i^*}] \tag{4.34}$$

satisfy input and state constrains (4.16b)-(4.16e). Therefore, (4.33)-(4.34) is a feasible solution for the LMPC (4.16) and (4.18) at time $t + 1$.

We showed that at the jth iteration, $\forall j \geq 1$, *(i):* the LMPC is feasible at time $t = 0$ and *(ii):* if the LMPC is feasible at time t, then the LMPC is feasible at time $t + 1$. Thus, we conclude by induction that the LMPC in (4.16) and (4.18) is feasible $\forall j \geq 1$ and $t \geq 0$.

Next we use the fact the Problem (4.16) is time-invariant at each iteration j and we replace $J^{\text{LMPC},j}_{t\to t+N}(\cdot)$ with $J^{\text{LMPC},j}_{0\to N}(\cdot)$. In order to show the asymptotic stability of x_F we have to show that the optimal cost, $J^{\text{LMPC},j}_{0\to N}(\cdot)$, is a Lyapunov function for the equilibrium point x_F of the closed loop system (4.1) and (4.18) [15]. Continuity of $J^{\text{LMPC},j}_{0\to N}(\cdot)$ can be shown as in [73]. From (4.1), $J^{\text{LMPC},j}_{0\to N}(x) \succ 0 \ \forall \ x \in \mathbb{R}^n \setminus \{x_F\}$ and $J^{\text{LMPC},j}_{0\to N}(x_F) = 0$. Thus, we need to show that $J^{\text{LMPC},j}_{0\to N}(\cdot)$ is decreasing along the closed loop trajectory.
From (4.32) we have $x^{*,j}_{t+1|t} = x^j_{t+1}$, which implies that

$$J^{\text{LMPC},j}_{0\to N}(x^*_{t+1|t}) = J^{\text{LMPC},j}_{0\to N}(x^j_{t+1}). \tag{4.35}$$

Given the optimal input sequence and the related optimal trajectory in (4.17), the optimal cost is given by

$$\begin{aligned}
J^{\text{LMPC},j}_{0\to N}(x^j_t) &= \min_{u_{t|t},\dots,u_{t+N-1|t}} \left[\sum_{k=0}^{N-1} h(x_{k|t}, u_{k|t}) + Q^{j-1}(x_{N|t}) \right] = \\
&= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + J^{i^*}_{t^*\to\infty}(x^{*,j}_{t+N|t}) = \\
&= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + \sum_{k=t^*}^{\infty} h(x^{i^*}_k, u^{i^*}_k) = \\
&= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + h(x^{i^*}_{t^*}, u^{i^*}_{t^*}) \ + Q^{j-1}(x^{i^*}_{t^*+1}) \geq \\
&\geq h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + J^{\text{LMPC},j}_{0\to N}(x^{*,j}_{t+1|t}),
\end{aligned} \tag{4.36}$$

where $x^{i^*}_{t^*}$ is defined in (4.13).

Finally, from equations (4.18), (4.30) and (4.35)-(4.36) we conclude that the optimal cost is a decreasing Lyapunov function along the closed loop trajectory,

$$J^{\text{LMPC},j}_{0\to N}(x^j_{t+1}) - J^{\text{LMPC},j}_{0\to N}(x^j_t) \leq -h(x^j_t, u^j_t) < 0, \forall \ x^j_t \in R^n \setminus \{x_F\}, \ \forall \ u^j_t \in R^m \setminus \{0\} \tag{4.37}$$

Equation (4.37), the positive definitiveness of $h(\cdot)$ and the continuity of $J^{\text{LMPC},j}_{0\to N}(\cdot)$ imply that x_F is asymptotically stable. ■

4.7.2 Proof of Theorem 2

First, we find a lower bound on the jth iteration cost $J^j_{0\to\infty}(\cdot), \ \forall \ j > 0$. Consider the realized state and input sequence (4.3) at the jth iteration, which collects the first element of the

optimal state and input sequence to the LMPC (4.16) and (4.18) at time t, $\forall t \geq 0$, as shown in (4.30). By the definition of the iteration cost in (4.10), we have

$$\begin{aligned} J^{j-1}_{0\to\infty}(x_S) &= \sum_{t=0}^{\infty} h(x_t^{j-1}, u_t^{j-1}) = \\ &= \sum_{t=0}^{N-1} h(x_t^{j-1}, u_t^{j-1}) + \sum_{t=N}^{\infty} h(x_t^{j-1}, u_t^{j-1}) \geq \\ &\geq \sum_{t=0}^{N-1} h(x_t^{j-1}, u_t^{j-1}) + Q^{j-1}(x_N^{j-1}) \geq \\ &\geq \min_{u_0,\ldots,u_{N-1}} \left[\sum_{k=0}^{N-1} h(x_k, u_k) + Q^{j-1}(x_N) \right] = J^{\text{LMPC},j}_{0\to N}(x_0^j). \end{aligned} \tag{4.38}$$

Then we notice that, at the jth iteration, the optimal cost of the LMPC (4.16) and (4.18) at $t = 0$, $J^{\text{LMPC},j}_{0\to N}(x_0^j)$, can be upper bounded along the realized trajectory (4.3) using (4.37)

$$\begin{aligned} J^{\text{LMPC},j}_{0\to N}(x_0^j) &\geq h(x_0^j, u_0^j) + J^{\text{LMPC},j}_{0\to N}(x_1^j) \quad \geq \\ &\geq h(x_0^j, u_0^j) + h(x_1^j, u_1^j) + J^{\text{LMPC},j}_{0\to N}(x_2^j) \quad \geq \\ &\geq \lim_{t\to\infty} \left[\sum_{k=0}^{t-1} h(x_k^j, u_k^j) + J^{\text{LMPC},j}_{0\to N}(x_t^j) \right]. \end{aligned} \tag{4.39}$$

From Theorem 1 x_F is asymptotically stable for the closed loop system (4.1) and (**??**) (i.e. $\lim_{t\to\infty} x_t^j = x_F$), thus by continuity of $h(\cdot,\cdot)$

$$\lim_{t\to\infty} J^{\text{LMPC},j}_{0\to N}(x_t) = J^{\text{LMPC},j}_{0\to N}(x_F) = 0. \tag{4.40}$$

From equations (4.39)-(4.40)

$$J^{\text{LMPC},j}_{0\to N}(x_0^j) \geq \sum_{k=0}^{\infty} h(x_k^j, u_k^j) = J^j_{0\to\infty}(x_S), \tag{4.41}$$

and finally from equations (4.38) and (4.41) we conclude that

$$J^{j-1}_{0\to\infty}(x_S) \geq J^{\text{LMPC},j}_{0\to N}(x_0^j) \geq J^j_{0\to\infty}(x_S), \tag{4.42}$$

thus the iteration cost is non-increasing. ■

4.7.3 Proof of Theorem 3

By assumption, system (4.1) in closed loop with the LMPC controller (4.16) and (4.18) converges to a steady state trajectory $\mathbf{x}^\infty$. This implies that both the sampled safe set SS^j and the terminal cost $Q^j(\cdot)$ converge at steady state, i.e, for $j \to \infty$, $\mathbf{x^j} \to \mathbf{x}^\infty$, $SS^j \to SS^\infty$ and $Q^j(\cdot) \to Q^\infty(\cdot)$. From (4.37) we have that

$$\begin{aligned} J^{\text{LMPC},\infty}_{0\to N}(x^\infty_t) &\geq h(x^\infty_t, u^\infty_t) + J^{\text{LMPC},\infty}_{0\to N}(x^\infty_{t+1}) \geq \\ &\geq h(x^\infty_t, u^\infty_t) + h(x^\infty_{t+1}, u^\infty_{t+1}) + J^{\text{LMPC},\infty}_{0\to N}(x^\infty_{t+2}) \geq \\ &\geq \left[\sum_{k=0}^{T-1} h(x^\infty_{t+k}, u^\infty_{t+k}) + sum^\infty_{k=T} h(x^\infty_{t+k}, u^\infty_{t+k})\right] \ \forall\ T > 0. \end{aligned} \tag{4.43}$$

From definition (4.6), we have that $\mathbf{x}^\infty \in \mathcal{SS}^\infty$. In equation (4.43), pick $T = N$ and from (4.43) we have:

$$J^{\text{LMPC},\infty}_{0\to N}(x^\infty_t) \geq \sum_{k=0}^{N-1} h(x^\infty_{t+k}, u^\infty_{t+k}) + Q^\infty(x^\infty_{t+N}). \tag{4.44}$$

From (4.44) we conclude that the cost associated with the feasible state and input trajectory

$$\begin{aligned} \mathbf{x}^\infty_{t:t+N} &= [x^\infty_t,\ x^\infty_{t+1},\ ...,\ x^\infty_{t+N}] \\ \mathbf{u}^\infty_{t:t+N} &= [u^\infty_t,\ u^\infty_{t+1},\ ...,\ u^\infty_{t+N-1}] \end{aligned} \tag{4.45}$$

is a lower bound of the optimal cost $J^{\text{LMPC},\infty}_{0\to N}(x^\infty_t)$. Therefore, $(\mathbf{x}^\infty_{t:t+N},\ \mathbf{u}^\infty_{t:t+N})$ is an optimal solution to the LMPC (4.16)-(4.18) for any t and for $j \to \infty$.

From remark 5 and from the above results, we have that $(\mathbf{x}^\infty_{t:t+N}, \mathbf{u}^\infty_{t:t+N})$ is an optimal solution to the optimal control problem defined in (4.24) with initial condition $x_t = x^\infty_t$ and $T = N$. The corresponding optimal cost is $\tilde{J}^*_{t\to t+N}(x^\infty_t)$.

Next, we prove that $\mathbf{x}^\infty_{t:t+N+1}$ and $\mathbf{u}^\infty_{t:t+N+1}$ is the optimal solution to the finite time optimal control problem (4.24) with initial condition $x_t = x^\infty_t$ and $T = N + 1$. The corresponding optimal cost is $\tilde{J}^*_{t\to t+N+1}(x^\infty_t)$. From time-invariance we focus on the case $t = 0$ and refer to Problem (4.24) with initial condition $x_0 = x^\infty_0$ and $T = N + 1$ as $J^*_{0\to N+1}(x^\infty_0)$.
We proceed by contradiction and assume that the optimal solution to problem $J^*_{0\to N+1}(x^\infty_0)$ is $(\tilde{x}^\infty_{0:N+1},\ \tilde{u}^\infty_{0:N+1})$ different from $(\mathbf{x}^\infty_{0:N+1}, \mathbf{u}^\infty_{0:N+1})$.

Define N feasible trajectories, for $\alpha \in (0, 1)$,

$$\hat{x}^{i,\infty}_{0:N+1} = [x^\infty_0, \ldots, x^\infty_{i-1}, \alpha\tilde{x}^\infty_i + (1-\alpha)x^\infty_i, x^\infty_{i+1}, \ldots, x^\infty_{N+1}] \tag{4.46}$$

with $i = [1, \ldots, N]$. By assumption $x^\infty_k \in \text{Int}(\text{Pre}(x^\infty_{k+1}))$ and $x^\infty_{k+1} \in \text{Int}(\text{Succ}(x^\infty_k))$ for all $k \geq 0$. This implies that there exists an $\alpha > 0$ such that the trajectory $\hat{x}^{i,\infty}_{0:N+1}$ and its related input sequence $\hat{u}^{i,\infty}_{0:N+1}$ are feasible for problem $J^*_{0\to N+1}(x^\infty_0)$.

For easier readability we introduce the function $J_{0\to N+1}(\cdot)$ which evaluates the cost of the $N+1$-steps trajectory:

$$J_{0\to N+1}(\mathbf{x}^{\infty}_{0:N+1}) = \sum_{k=0}^{N} h(x^{\infty}_k, u^{\infty}_k) + Q(x^{\infty}_{N+1}). \tag{4.47}$$

We notice that, by optimality of $(\mathbf{x}^{\infty}_{t:t+N}, \mathbf{u}^{\infty}_{t:t+N})$ for all $t \geq 0$, we have

$$J_{0\to N}(\hat{x}^{i,\infty}_{0:N}) > J_{0\to N}(\mathbf{x}^{\infty}_{0:N}), \quad \forall i \in \{1,\dots,N-1\}, \tag{4.48a}$$

$$J_{1\to N+1}(\hat{x}^{i,\infty}_{1:N+1}) > J_{1\to N+1}(\mathbf{x}^{\infty}_{1:N+1}), \quad \forall i \in \{2,\dots,N\}. \tag{4.48b}$$

From (4.48a) we have

$$\sum_{k=0}^{N-1} h(\hat{x}^{i,\infty}_k, \hat{u}^{i,\infty}_k) + Q(\hat{x}^{i,\infty}_N) > \sum_{k=0}^{N-1} h(x^{\infty}_k, u^{\infty}_k) + Q(x^{\infty}_N) \ \forall i \in \{1,\dots,N-1\}. \tag{4.49}$$

Moreover, we know that $\hat{x}^{i,\infty}_N = x^{\infty}_N$ and $\hat{u}^{i,\infty}_N = u^{\infty}_N \ \forall i \in \{1,\dots,N-1\}$, therefore by definition of the terminal cost (4.11) and from (4.49) we have

$$\sum_{k=0}^{N} h(\hat{x}^{i,\infty}_k, \hat{u}^{i,\infty}_k) + Q(\hat{x}^{i,\infty}_{N+1}) > \sum_{k=0}^{N} h(x^{\infty}_k, u^{\infty}_k) + Q(x^{\infty}_{N+1}) \ \forall i \in \{1,\dots,N-1\}, \tag{4.50}$$

which implies

$$J_{0\to N+1}(\hat{x}^{i,\infty}_{0:N+1}) > J_{0\to N+1}(\mathbf{x}^{\infty}_{0:N+1}), \quad \forall i \in \{1,\dots,N-1\}. \tag{4.51}$$

Moreover, from the fact that $\hat{x}^{i,\infty}_0 = x^{\infty}_0$ and $\hat{u}^{i,\infty}_0 = u^{\infty}_0, \ \forall i = \{2,\dots,N\}$ and from (4.48b) we have

$$h(\hat{x}^{i,\infty}_0, \hat{u}^{i,\infty}_0) + J_{1\to N+1}(\hat{x}^{i,\infty}_{1:N+1}) > h(x^{\infty}_0, u^{\infty}_0) + J_{1\to N+1}(\mathbf{x}^{\infty}_{1:N+1}), \ \forall i \in \{2,\dots,N\}. \tag{4.52}$$

From (4.51) and (4.52) we conclude that

$$J_{0\to N+1}(\hat{x}^{i,\infty}_{0:N+1}) > J_{0\to N+1}(\mathbf{x}^{\infty}_{0:N+1}), \quad \forall i \in \{1,\dots,N\}. \tag{4.53}$$

Define the trajectory $\bar{x}^{\infty}_{0:N+1}$ as convex combination of $\mathbf{x}^{\infty}_{0:N+1}$ and the trajectories in (4.46),

$$\bar{x}^{\infty}_{0:N+1} = \sum_{i=1}^{N} \frac{1}{N}\Big(\frac{1}{2}\hat{x}^{i,\infty}_{0:N+1} + \frac{1}{2}\mathbf{x}^{\infty}_{0:N+1}\Big). \tag{4.54}$$

From (4.46) we have that $\bar{x}^{\infty}_{0:N+1}$ can be expressed also as a convex combination of the optimal trajectory $\tilde{x}^{\infty}_{0:N+1}$ and $\mathbf{x}^{\infty}_{0:N+1}$,

$$\bar{x}^{\infty}_{0:N+1} = \frac{\alpha}{2N}\tilde{x}^{\infty}_{0:N+1} + \frac{2N-\alpha}{2N}\mathbf{x}^{\infty}_{0:N+1}. \tag{4.55}$$

Concluding from (4.54) and Assumption 2, we have that

$$J_{0\to N+1}(\mathbf{x}^{\infty}_{0:N+1}) < J_{0\to N+1}(\bar{x}^{\infty}_{0:N+1}) < J_{0\to N+1}(\hat{x}^{k,\infty}_{0:N+1}) \tag{4.56}$$

where $k = \arg\max_{i\in[1,\dots,N]} J_{0\to N+1}(\hat{x}^{i,\infty}_{0:N+1})$.
Furthermore, from (4.55) and Assumption 2, we have that

$$J_{0\to N+1}(\tilde{x}^{\infty}_{0}) < J_{0\to N+1}(\bar{x}^{\infty}_{0}) < J_{0\to N+1}(\mathbf{x}^{\infty}_{0:N+1}). \tag{4.57}$$

Finally, from (4.56) and (4.57) we have a contradiction and we conclude that $(\mathbf{x}^{\infty}_{0:N+1}, \mathbf{u}^{\infty}_{0:N+1})$ is the optimal solution of the finite time optimal control problem (4.24) with initial condition $x_t = x^{\infty}_t$ and $T = N + 1$. The above procedure can be iterated for $T = N + 2$, $T = N + 3,\dots$ which proves the Theorem. ■

4.7.4 Convex LMPC Properties

In this Section, the properties of $\mathcal{CS}^j$ and $Q^j_c(\cdot)$ are used to show recursive feasibility and asymptotic stability of the equilibrium point x_F.

4.7.4.1 Recursive Feasibility and Stability

We show that for deterministic liner system

$$x_{t+1} = Ax_t + Bu_t, \tag{4.58}$$

the convex LMPC guarantees recursive constraint satisfaction and stability goal state x_F for the closed-loop system.

Theorem 4 *Consider system* (4.58) *controlled by the LMPC controller* (4.19) *and* (4.20). *Let* $\mathcal{CS}^j$ *be the convex safe set at iteration* j *as defined in (4.8). Let Assumption 1 hold, then the LMPC* (4.19) *and* (4.20) *is feasible* $\forall\ t \in \mathbb{Z}_{0+}$ *and iteration* $j \geq 1$. *Moreover, the equilibrium point* x_F *is asymptotically stable for the closed loop system* (4.58) *and* (4.20) *at every iteration* $j \geq 1$.

Proof *The proof follows from standard MPC arguments. By assumption* $\mathcal{CS}^0$ *is non empty. From (4.8) we have that* $\mathcal{CS}^0 \subseteq \mathcal{CS}^{j-1}\ \forall j \geq 1$, *and consequently* $\mathcal{CS}^{j-1}$ *is a non empty set. In particular, there exists a trajectory* $\mathbf{x}^0 \in \mathcal{CS}^0 \subseteq \mathcal{CS}^{j-1}$. *From (4.4) we know that* $x^j_0 = x_S\ \forall j \geq 0$. *At time* $t = 0$ *of the* j*-th iteration the* N *steps trajectory*

$$[x^0_0,\ x^0_1,\ \dots, x^0_N] \in \mathcal{CS}^{j-1}, \tag{4.59}$$

and the related input sequence,

$$[u^0_0,\ u^0_1,\ \dots, u^0_{N-1}], \tag{4.60}$$

satisfy input and state constrains (4.19b)-(4.19f). *Therefore (4.59)-(4.60) is a feasible solution to the LMPC (4.19) and (4.20) at* $t = 0$ *of the* j*-th iteration.*
Assume that at time t *of the* j*-th iteration the LMPC (4.19) and (4.20) is feasible and let* $\mathbf{x}^{*,j}_{t:t+N|t}$ *and* $\mathbf{u}^{*,j}_{t:t+N|t}$ *be the optimal trajectory and input sequence. From (4.19b) and (4.20) the realized state and input at time* t *of the* j*-th iteration are given by*

$$\begin{aligned} x^j_t &= x^{*,j}_{t|t}, \\ u^j_t &= u^{*,j}_{t|t}. \end{aligned} \tag{4.61}$$

Moreover, the terminal constraint (4.19e) enforces $x^{*,j}_{t+N|t} \in \mathcal{CS}^{j-1}$ *and, from* (4.14),

$$x^{*,j}_{t+N|t} = \sum_{k=0}^{j-1} \sum_{t=0}^{\infty} \lambda^{*,k}_t x^k_t. \tag{4.62}$$

We define

$$\bar{u} = \sum_{k=0}^{j-1} \sum_{t=0}^{\infty} \lambda^{*,k}_t u^k_t, \quad \in \mathcal{U}, \tag{4.63}$$

and

$$\bar{x} = A x^{*,j}_{t+N|t} + B\bar{u} = \sum_{k=0}^{j-1} \sum_{t=0}^{\infty} \lambda^{*,k}_t \Big(A x^k_t + B u^k_t \Big) = \sum_{k=0}^{j-1} \sum_{t=0}^{\infty} \lambda^{*,k}_t x^k_{t+1} \in \mathcal{CS}^{j-1}. \tag{4.64}$$

Since the state update in (4.58) and (4.19b) are assumed identical we have that

$$x^j_{t+1} = x^{*,j}_{t+1|t}. \tag{4.65}$$

At time $t+1$ *of the* j*-th iteration the input sequence and the related feasible state trajectory*

$$[u^{*,j}_{t+1|t},\ u^{*,j}_{t+2|t},\ ...,\ u^{*,j}_{t+N-1|t},\ \bar{u}], \tag{4.66a}$$

$$[x^{*,j}_{t+1|t},\ x^{*,j}_{t+2|t},\ ...,\ x^{*,j}_{t+N-1|t},\ x^{*,j}_{t+N|t},\ \bar{x}] \tag{4.66b}$$

satisfy input and state constrains (4.19b)-(4.19f). *Therefore, (4.66) is a feasible solution for the LMPC (4.19) and (4.20) at time* $t+1$.
We showed that at the j*-th iteration,* $\forall j \geq 1$, (i): *the LMPC is feasible at time* $t = 0$ *and* (ii): *if the LMPC is feasible at time* t*, then the LMPC is feasible at time* $t+1$. *Thus, we conclude by induction that the LMPC in (4.19) and (4.20) is feasible* $\forall j \geq 1$ *and* $t \in \mathbb{Z}_{0+}$.

Next we use the fact the Problem (4.19) is time-invariant at each iteration j *and we replace* $J^{\text{LMPC},j}_{c,t\to t+N}(\cdot)$ *with* $J^{\text{LMPC},j}_{c,0\to N}(\cdot)$. *In order to show the asymptotic stability of* x_F *we have to show that the optimal cost,* $J^{\text{LMPC},j}_{c,0\to N}(\cdot)$*, is a Lyapunov function for the equilibrium point* x_F *of the closed loop system (4.58) and (4.20) [15]. Continuity of* $J^{\text{LMPC},j}_{c,0\to N}(\cdot)$ *can be shown as in [73]. Moreover from (4.19a),* $J^{\text{LMPC},j}_{c,0\to N}(x) \succ 0\ \forall\ x \in \mathbb{R}^n \setminus \{x_F\}$ *and* $J^{\text{LMPC},j}_{c,0\to N}(x_F) = 0$. *Thus, we*

need to show that $J^{LMPC,j}_{c,0\to N}(\cdot)$ is decreasing along the closed loop trajectory. From (4.65) we have $x^{*,j}_{t+1|t} = x^j_{t+1}$, which implies that

$$J^{LMPC,j}_{c,0\to N}(x^*_{t+1|t}) = J^{LMPC,j}_{c,0\to N}(x^j_{t+1}). \tag{4.67}$$

Given the optimal input sequence and the related optimal trajectory and the definition of the $Q^{j-1}_c(\cdot)$ (4.14), the optimal cost is given by

$$\begin{aligned} J^{LMPC,j}_{c,0\to N}(x^j_t) &= \min_{u_{t|t},\dots,u_{t+N-1|t}} \left[\sum_{k=0}^{N-1} h(x_{k|t}, u_{k|t}) + Q^{j-1}_c(x_{N|t}) \right] = \\ &= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + Q^{j-1}_c(x^{*,j}_{t+N|t}) = \\ &= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + \sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t \sum_{l=0}^{\infty} h(x^k_{t+l}, u^k_{t+l}). \end{aligned} \tag{4.68}$$

We can further simplify the above expression using (4.14), (4.62)-(4.64) and the fact that $h(\cdot,\cdot)$ is jointly convex in the arguments,

$$\begin{aligned} J^{LMPC,j}_{c,0\to N}(x^j_t) &= h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + \sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t h(x^k_t, u^k_t) \\ &\qquad + \sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t \sum_{l=1}^{\infty} h(x^k_{t+l}, u^k_{t+l}) \\ &\geq h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + h\Big(\sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t x^k_t, \sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t u^k_t \Big) + \\ &\qquad + \sum_{k=0}^{j-1}\sum_{t=0}^{\infty} \lambda^{*,k}_t J^k_{t\to\infty}(x^k_{t+1}) \\ &\geq h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + \sum_{k=1}^{N-1} h(x^{*,j}_{t+k|t}, u^{*,j}_{t+k|t}) + h(x^{*,j}_{t+N|t}, \bar{u}) + Q^{j-1}_c(\bar{x}) \\ &\geq h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + J^{LMPC,j}_{c,0\to N}(x^{*,j}_{t+1|t}). \end{aligned} \tag{4.69}$$

Note that, in the above derivation, we used the fact that $\bar{\lambda}^k_0 = 0$ and $\bar{\lambda}^k_{t+1} = \lambda^{*,k}_t$, $\forall k \in \{0, j-1\}$, $t \in \mathbb{Z}_{0+}$ is a feasible solution to problem (8.13) and therefore $\sum_{t=0}^{\infty} \lambda^{*,k}_t J^k_{t\to\infty}(x^k_{t+1})$ is a upper bound for $Q^{j-1}_c(\bar{x})$. Finally, from equations (4.20), (4.61) and (4.67)-(4.69) we conclude that the optimal cost is a decreasing Lyapunov function along the closed loop trajectory,

$$J^{LMPC,j}_{c,0\to N}(x^j_{t+1}) - J^{LMPC,j}_{c,0\to N}(x^j_t) \leq -h(x^j_t, u^j_t) < 0, \forall\ x^j_t \in R^n \setminus \{x_F\} \tag{4.70}$$

Equation (4.70), the positive definitiveness of $h(\cdot,\cdot)$ and the continuity of $J^{LMPC,j}_{c,0\to N}(\cdot)$ imply that x_F is asymptotically stable.

4.7.4.2 Performance Improvement and Convergence Properties

In this section we show two results. First, the jth iteration cost $J^j_{0\to\infty}(\cdot)$ does not increase as j increases. Second, if the closed-loop system converges to a steady state behavior in the iteration domain, then the steady state trajectory is a local optimal solution to the infinite horizon control problem (4.5). In the following, we use the fact the Problem (4.19) is time-invariant at each iteration j and we replace $J^{\text{LMPC},j}_{c,t\to t+N}(\cdot)$ with $J^{\text{LMPC},j}_{c,0\to N}(\cdot)$.

Theorem 5 *Consider system (4.58) in closed loop with the LMPC controller (4.19) and (4.20). Let $\mathcal{CS}^j$ be the convex safe set at the j-th iteration as defined in (4.8). Let Assumption 1 hold, then the iteration cost $J^j_{0\to\infty}(\cdot)$ does not increase with the iteration index j.*

Proof *Notice from the Follows from* (4.69) *that $J^{LMPC,j}_{c,0\to N}(x^j_t) \geq h(x^{*,j}_{t|t}, u^{*,j}_{t|t}) + J^{LMPC,j}_{c,0\to N}(x^{*,j}_{t+1|t})$. Therefore, the proof follows as in Theorem 2.*

Theorem 6 *Consider system (4.58) in closed loop with the LMPC controller (4.19) and (4.20) with $N > 1$. Let $\mathcal{CS}^j$ be the sampled safe set at the jth iteration as defined in (4.8). Let Assumptions 1-2 hold and assume that the LMPC controller (4.19) and (4.20) converges to the steady state input $\mathbf{u}^\infty = \lim_{j\to\infty} \mathbf{u}^j$ and the steady state trajectory $\mathbf{x}^\infty = \lim_{j\to\infty} \mathbf{x}^j$, for iteration $j \to \infty$. Denote $\text{Int}(\mathcal{S})$ as the interior of the set $\mathcal{S}$, and recall the definition of one-step predecessor $\text{Pre}(\cdot)$ and successor $\text{Succ}(\cdot)$ sets from Chapter 2. If $x^\infty_k \in \text{Int}(\text{Pre}(x^\infty_{k+1}))$ and $x^\infty_{k+1} \in \text{Int}(\text{Succ}(x^\infty_k))$ for all $k \geq 0$, then $(\mathbf{x}^\infty_{t:t+T}, \mathbf{u}^\infty_{t:t+T})$ is the optimizer of the finite horizon optimal control problem (4.24) with initial condition $x_t = x^\infty_t$ for all $t \geq 0$ and for all $T > 0$.*

Proof *The proof follows as in Theorem 3.*

4.7.5 Local LMPC Properties

Finally, we describe the properties of the local LMPC policy (4.21) in closed-loop with system (4.23). In particular, we show the local LMPC strategy guarantee recursive constraint satisfaction, closed-loop stability and performance improvement.

Theorem 7 *Consider system (4.1) controlled by the local LMPC* (4.21) *and* (4.23)*. Let $\mathcal{LS}^j(z_t)$ be the local sampled safe set at iteration j as defined in (4.9), and z_t the vector from* (4.22)*. Let Assumption 1 hold, then the LMPC (4.21) and (4.23) is feasible for all $t \geq 0$ and iteration $j \geq 1$. Moreover, the equilibrium point x_F is asymptotically stable for the closed loop system (4.1),* (4.21) *and (4.23) at every iteration $j \geq 1$.*

Proof *The proof follows from standard MPC arguments.*
By definition $x_N^j \in \mathcal{LS}^0(z_0^j)$. *Therefore, at time* $t = 0$ *of the* j*th iteration the* N *steps trajectory*

$$\mathbf{x}_{0:N}^0 = [x_0^0,\ x_1^0,\ ..., x_N^0], \tag{4.71}$$

and the related input sequence,

$$[u_0^0,\ u_1^0,\ ..., u_{N-1}^0], \tag{4.72}$$

satisfy input and state constrains (4.21b)-(4.21e). Therefore (4.71)-(4.72) is a feasible solution to the local LMPC (4.21) and (4.23) at $t = 0$ *of the* j*th iteration.*
Assume that at time t *of the* j*th iteration the LMPC (4.21) and (4.23) is feasible and let* $\mathbf{x}_{t:t+N|t}^{*,j}$ *and* $\mathbf{u}_{t:t+N|t}^{*,j}$ *be the optimal trajectory and input sequence. From (4.21b) and (4.23) the realized state and input at time* t *of the* j*th iteration are given by*

$$\begin{aligned} x_t^j &= x_{t|t}^{*,j}, \\ u_t^j &= u_{t|t}^{*,j}. \end{aligned} \tag{4.73}$$

The terminal constraint (4.21d) *enforces* $x_{t+N|t}^{*,j} \in \mathcal{LS}^{j-1}(z_t^j)$ *and, from (4.9),*

$$Q_l^{j-1}(x_{t+N|t}^{*,j}, z_t) = J_{t^*\to\infty}^{i^*}(x_{t+N|t}^{*,j}) = \sum_{k=t^*}^{\infty} h(x_k^{i^*}, u_k^{i^*}). \tag{4.74}$$

Note that, by the definition (4.22), $x_{t^*+1}^{i^*} = f(x_{t^*}^{i^*}, u_{t^*}^{i^*}) = z_{t+1}^j$ *and* $x_{t^*}^{i^*} = x_{t+N|t}^{*,j}$. *Since the state update in* (4.1) *and* (4.16b) *are assumed identical we have that*

$$x_{t+1}^j = x_{t+1|t}^{*,j}. \tag{4.75}$$

At time $t+1$ *of the* j*th iteration the input sequence*

$$[u_{t+1|t}^{*,j},\ u_{t+2|t}^{*,j},\ ...,\ u_{t+N-1|t}^{*,j},\ u_{t^*}^{i^*}], \tag{4.76}$$

and the related feasible state trajectory

$$[x_{t+1|t}^{*,j},\ x_{t+2|t}^{*,j},\ ...,\ x_{t+N-1|t}^{*,j},\ x_{t^*}^{i^*},\ x_{t^*+1}^{i^*} \in \mathcal{LS}^{j-1}(z_{t+1}^j)] \tag{4.77}$$

satisfy input and state constrains (4.21b)-(4.21e). *Therefore, (4.76)-(4.77) is a feasible solution for the LMPC (4.21) and (4.23) at time* $t+1$.
We showed that at the j*th iteration,* $\forall j \geq 1$, (i): *the local LMPC is feasible at time* $t = 0$ *and* (ii): *if the LMPC is feasible at time* t*, then the LMPC is feasible at time* $t+1$. *Thus, we conclude by induction that the local LMPC in (4.21) and (4.23) is feasible* $\forall j \geq 1$ *and* $t \geq 0$.

The stability of the closed-loop system can be shown as in Theorem 9. In particular, we can exploit feasibility of (4.76)-(4.77) *and the definition* (4.9) *to show that the local LMPC cost* $J_{l,t\to t+N}^{LMPC,j}(\cdot)$ *is a control Lyapunov function for the closed-loop trajectory.*

Theorem 8 *Consider system (4.1) in closed loop with the LMPC controller (4.21) and (4.23). Let $\mathcal{LS}^j(z_t)$ be the local sampled safe set at iteration j as defined in (4.9), and z_t the vector from* (4.22)*. Let Assumption 1 hold, then the iteration cost $J^j_{0\to\infty}(\cdot) = \sum_{t=0}^{\infty} h(x^j_t, u^j_t)$ does not increase with the iteration index j.*

Proof *The stability of the closed-loop system can be shown as in Theorem 2. In particular, we can exploit feasibility of* (4.76)-(4.77) *and the definition* (4.9) *to show that the local LMPC cost $J^{LMPC,j}_{l,t\to t+N}(x^j_t) \leq h(x^j_t, u^j_t) + J^{LMPC,j}_{l,t\to t+N}(x^j_{t+1})$.*

Chapter 5

Time-Varying LMPC for Time Optimal Control Problems

In time optimal control problems, the goal of the controller is to steer the system from the starting point x_S to the terminal point x_F in minimum time, while satisfying state and input constraints. These problems have been studied since the 1950s [74, 75, 76, 77] and it was shown that the optimal input strategy is a piece-wise function which saturates the input constraints [74, 75, 76]. Furthermore, while investigating the solution to time optimal control problems, researches formalized the maximum principle which describes the first order necessary optimality conditions [78, 72]. For linear systems, time optimal control problems can be solved applying the maximum principle. However, for nonlinear systems the optimality conditions are hard to solve, as those are described by a two boundary value problem for a system of nonlinear differential equations [72].

In this chapter, we focus on nonlinear time optimal control problems, and we design Time-Varying Learning Model Predictive Controllers (LMPC) which guarantees recursive constraint satisfaction, convergence in finite time and iterative performance improvement. Compared with the previous chapter, the safe set and Q-function are time varying. Furthermore, we show that convexifing the time varying safe set and Q-function allows us to reduce the computational cost while guaranteeing safety and performance improvement for a class of nonlinear system and convex constraints. Finally, we illustrate the effectiveness of the proposed strategies on minimum time obstacle avoidance and racing examples.

5.1 Problem Formulation

Consider the nonlinear system

$$x_{t+1}^j = f(x_t^j, u_t^j), \tag{5.1}$$

where at time t of the jth iteration the state $x_t^j \in \mathbb{R}^n$ and the input $u_t^j \in \mathbb{R}^d$. Furthermore, the system is subject to the following state and input constraints

$$x_t^j \in \mathcal{X} \text{ and } u_t^j \in \mathcal{U}, \forall t \geq 0, \forall j \geq 0. \tag{5.2}$$

The goal of the controller is solve the following minimum time optimal control problem

$$\begin{aligned} \min_{T, u_0^j, \ldots, u_{T-1}^j} \quad & \sum_{t=0}^{T-1} 1 \\ \text{s.t.} \quad & x_{t+1}^j = f(x_t^j, u_t^j), \forall t = [0, \ldots, T-1] \\ & x_t^j \in \mathcal{X},\ u_t^j \in \mathcal{U}, \forall t = [0, \ldots, T-1] \\ & x_T^j = x_F, \\ & x_0^j = x_S \end{aligned} \tag{5.3}$$

where the goal state x_F is an unforced equilibrium point for system (5.1), i.e. $f(x_F, 0) = x_F$.

We propose to solve Problem (5.3) iteratively. In particular, at each iteration we drive the system from the starting point x_S to the terminal state x_F and we store the closed-loop trajectories. After completion of the jth iteration, these trajectories are used to synthesize a control policy for the next iteration $j+1$. We show that the proposed iterative design strategy guarantees recursive constraint satisfaction and iterative performance improvement. Next, we define the safe set and value function approximation which will be used in the controller design.

5.2 Safe Set and Value Function Approximation

At each jth iteration of the control task, we store the closed-loop trajectories and the associated input sequences. In particular, at the jth iteration we define the vectors

$$\begin{aligned} \mathbf{u}^j &= [u_0^j, \ldots,\ u_{T^j}^j], \\ \mathbf{x}^j &= [x_0^j, \ldots,\ x_{T^j}^j], \end{aligned} \tag{5.4}$$

where x_t^j and u_t^j are the state and input of system (5.1). In (5.4), T^j denotes the time at which the closed-loop system reached the terminal state, i.e. $x_{T^j} = x_F$.

5.2.1 Time Varying Safe Set

We use the stored data to build time varying safe sets, which will be used in the controller design to guarantee recursive constraint satisfaction. First, we define the time varying safe set at iteration j as

$$\mathcal{SS}_t^j = \bigcup_{i=0}^{j} \bigcup_{k=\delta_t^i}^{T^i} x_k^i, \tag{5.5}$$

where, for $T^{j,*} = \min_{k\in\{0,\dots,j\}} T^k$,

$$\delta_t^i = \max(t + T^i - T^{j,*}, 0). \tag{5.6}$$

Definition (5.6) implies that if at time t of the jth iteration $x_t^j = x_{\delta_t^i}^i$, then system (5.1) can be steered along the ith trajectory to reach x_F in $(T^{j,*} - t)$ time steps. Basically, at each time t the time varying safe set $\mathcal{SS}_t^j$ collects the stored states from which system (5.1) can reach the terminal state x_F in at most $(T^{j,*} - t)$ time steps. A representation of the time varying safe set for a two-dimensional system is shown in Figure 5.1. We notice that, by definition, if a state x_t^i belongs to $\mathcal{SS}_t^j$, then there exists a feasible control action $u_t^i \in \mathcal{U}$ which keeps the evolution of the nonlinear system (5.1) into the time varying safe set at the next time step $t+1$, i.e. $f(x_t^i, u_t^i) \in \mathcal{SS}_{t+1}^j$. This property will be used in the controller design to guarantee that state and input constraints (5.2) are recursively satisfied.

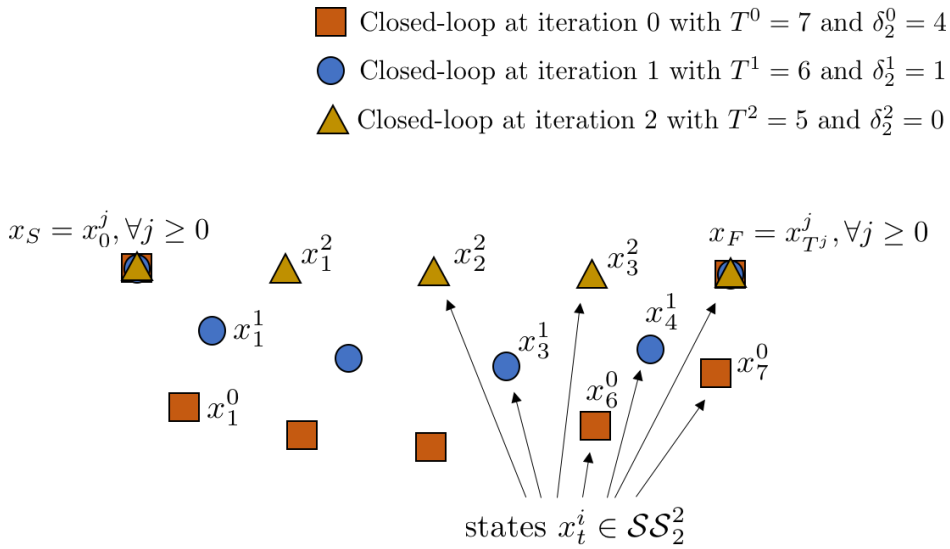


Figure 5.1: Representation of the time varying safe set $\mathcal{SS}_2^2$. We notice that just a subset of the stored states are used to define $\mathcal{SS}_2^2$. Furthermore, we notice that from all states $x_t^i \in \mathcal{SS}_2^2$ system (5.1) can be steered to x_F in at most $T^{j,*} - t = 2$ time steps.

Finally, at each time t the we define the local convex safe set as the convex hull of $\mathcal{SS}_t^j$ from (5.5),

$$\begin{aligned}\mathcal{CS}_t^j &= \text{Conv}\big(\mathcal{SS}_t^j\big) \\ &= \big\{x \in \mathbb{R}^n : \exists[\lambda_{\delta_t^0}^0, ..., \lambda_{T^j}^j] \geq 0, \sum_{i=0}^{j}\sum_{k=\delta_t^i}^{T^i} \lambda_k^i = 1, \sum_{i=0}^{j}\sum_{k=\delta_t^i}^{T^i} \lambda_k^i x_k^i = x\big\}.\end{aligned} \tag{5.7}$$

Later on we will show that for a class on nonlinear systems, if a state x_t^i belongs to $\mathcal{CS}_t^j$, then there exists a feasible control action $u_t^i \in \mathcal{U}$ which keeps the evolution of the nonlinear

system (5.1) into the convex safe set at the next time step $t+1$. For such class of nonlinear systems, $\mathcal{CS}_t^j$ can be used to synthesize controllers which guarantee state and input constraint satisfaction at all time instants.

Remark 7 *When the goal of the controller is to reach an invariant set $\mathcal{X}_F$ in minimum time, it is still possible to use the proposed iterative control strategy. In this case one should replace $x_{T^i}^i = x_F$ with $\mathcal{X}_F$ in definition* (5.5).

5.2.2 Time Varying Value Function Approximation

In this section, we show how to construct Q-functions which approximate the cost-to-go over the safe set and convex safe set. These functions will be used in the controller design to guarantee iterative performance improvement.

We define the cost-to-go associated with the stored state x_t^j from (5.4),

$$J_{t\to T^j}^j(x_t^j) = \sum_{k=t}^{T^j} \mathbb{1}_{x_F}(x_k^j), \tag{5.8}$$

where the indicator function

$$\mathbb{1}_{x_F}(x) = \begin{cases} 1 & \text{If } x_F \neq x \\ 0 & \text{Else} \end{cases}.$$

The above cost-to-go represents the time steps needed to steer system (5.1) from x_t^j to the terminal state x_F along the jth trajectory, and it is used to construct the function $Q_t^j(\cdot)$, defined over the safe set $\mathcal{SS}_t^j$,

$$\begin{aligned} Q_t^j(x) = \min_{\substack{i\in\{0,\dots,j\}\\ k\in\{\delta_t^i,\dots,T^i\}}} \quad & J_{k\to T^i}^i(x_k^i) \\ \text{s.t.} \quad & x = x_k^i \in \mathcal{SS}_t^j. \end{aligned} \tag{5.9}$$

The function $Q_t^j(\cdot)$ assigns to every point in the safe set $\mathcal{SS}_t^j$ from (5.5) the minimum cost-to-go along the stored trajectories from (5.4), i.e.

$$\forall x \in \mathcal{SS}_t^j, Q_t^j(x) = J_{k^*\to T^{(i^*)}}^{(i^*)}(x) = \sum_{k=k^*}^{T^{(i^*)}} \mathbb{1}_{x_F}\left(x_k^{(i^*)}\right)$$

where i^* and k^* are the minimizers in (5.9):

$$\begin{aligned} [i^*, k^*] = \operatorname*{argmin}_{\substack{i\in\{0,\dots,j\}\\ k\in\{\delta_t^i,\dots,T^i\}}} \quad & J_{k\to T^i}^i(x_k^i) \\ \text{s.t.} \quad & x = x_k^i \in \mathcal{SS}_t^j. \end{aligned} \tag{5.10}$$

Finally, we define the convex Q-function over the convex safe set $\mathcal{CS}_t^j$ from (5.7),

$$\begin{aligned} \bar{Q}_t^j(x) = \min_{[\lambda_{\delta_t^0}^0,\ldots,\lambda_{T^j}^j]\geq 0} \quad & \sum_{i=0}^{j}\sum_{k=\delta_t^i}^{T^i} \lambda_k^i J_{k\to T^i}^i(x_k^i) \\ \text{s.t.} \quad & \sum_{i=0}^{j}\sum_{k=\delta_t^i}^{T^i} \lambda_k^i x_k^i = x \\ & \sum_{i=0}^{j}\sum_{k=\delta_t^i}^{T^i} \lambda_k^i = 1. \end{aligned} \tag{5.11}$$

where δ_t^i is defined in (5.6). The convex Q-function $\bar{Q}_t^j(\cdot)$ is simply a piecewise-affine interpolation of Q-function from (5.9) over the convex safe set, as shown in Figure 5.2. In Section 5.4, we will show that the convex Q-function can be used to guarantee iterative performance improvement for a particular class on nonlinear systems.

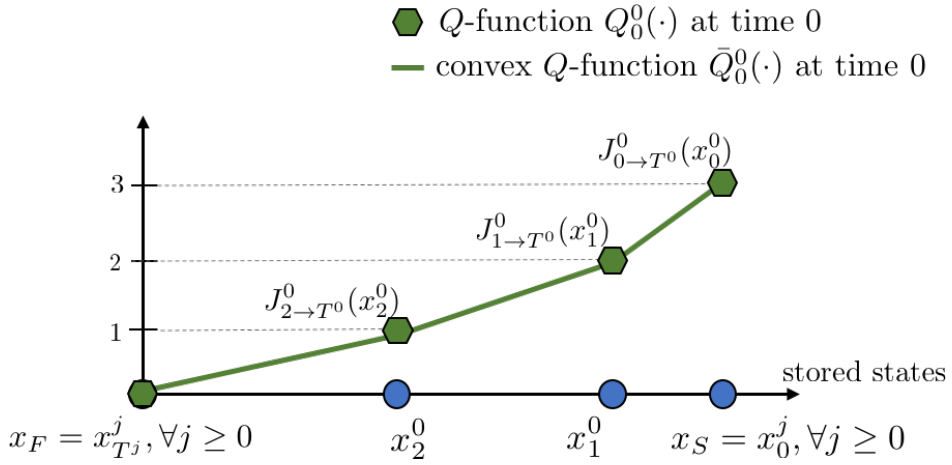


Figure 5.2: Representation of the Q-function $Q_0^0(\cdot)$ and convex Q-function $\bar{Q}_0^0(\cdot)$. We notice that the Q-function $Q_0^0(\cdot)$ is defined over a set of discrete data points, whereas the convex Q-function $\bar{Q}_0^0(\cdot)$ is defined over the convex safe set.

5.3 Control Design

In this section, we describe the controller design. We propose a Learning Model Predictive Control (LMPC) strategy for nonlinear systems which guarantees recursive constraint satisfaction and iterative performance improvement. Computing the control action from the LMPC policy is expensive. For this reason, we also present a relaxed LMPC policy, which allows us to reduce the computational cost and it guarantees recursive constraint satisfaction and performance improvement for a class of nonlinear systems.

5.3.1 LMPC: Mixed Integer Formulation

At each time t of the jth iteration the controller solves the following finite time optimal control problem,

$$\begin{aligned} J^{\text{LMPC},j}_{t\to t+N}(x^j_t) = \min_{\mathbf{U}^j_t} \quad & \left[\sum_{k=t}^{t+N-1} \mathbb{1}_{x_F}(x^j_{k|t}) + Q^{j-1}_{t+N}(x^j_{t+N|t}) \right] \\ \text{s.t.} \quad & x^j_{k+1|t} = f(x^j_{k|t}, u^j_{k|t}), \forall k = t, \cdots, t+N-1 \\ & x^j_{k|t} \in \mathcal{X}, u^j_{k|t} \in \mathcal{U}, \forall k = t, \cdots, t+N-1 \\ & x^j_{t+N|t} \in \mathcal{SS}^{j-1}_{t+N} \\ & x^j_{t|t} = x^j_t \end{aligned} \tag{5.12}$$

where $\mathbf{U}^j_t = [u^j_{t|t}, \ldots, u^j_{t+N-1|t}] \in \mathbb{R}^{d\times N}$. The solution to the above finite time optimal control problem steers system (5.1) from x^j_t to the time varying safe set $\mathcal{SS}^{j-1}_t$ while satisfying state, input and dynamic constraints. Let

$$\mathbf{U}^{j,*}_t = [u^{j,*}_{t|t}, \ldots, u^{j,*}_{t+N-1|t}] \tag{5.13}$$

be the optimal solution to (5.12) at time t of the jth iteration. Then, we apply to the system (5.1) the first element of the optimizer vector,

$$u^j_t = \pi^{\text{LMPC},j}_t(x^j_t) = u^{j,*}_{t|t}. \tag{5.14}$$

The finite time optimal control problem (5.12) is repeated at time $t+1$, based on the new state $x_{t+1|t+1} = x^j_{t+1}$, until the iteration is terminated when $x^j_{t+1} = x_F$.

Computing the control action from the LMPC policy (5.14) requires to solve a mixed-integer programming problem, as $\mathcal{SS}^{j-1}_t$ is a set of discrete states. In particular, the number of integer variables grows are more iterations are stored. In Section 5.5 we will show that the computational cost may be reduced synthesizing the LMPC policy (5.14) using a subset of the stored iterations and P data points per iteration. Finally, in the result section we will show that the number of data points used in the synthesis process affects the performance improvement at each iteration. Therefore there is a trade-off between the online computational burden and the number of iterations needed to reach desirable closed-loop performance.

5.3.2 Relaxed LMPC: Nonlinear Formulation

In this section, we present a relaxed LMPC constructed using the convex safe set $\mathcal{CS}^{j-1}_t$ in (5.7). At each time t of the jth iteration we solve the following finite time optimal control

problem

$$\begin{aligned}
\bar{J}^{\text{LMPC},j}_{t\to t+N}(x^j_t) = \min_{\mathbf{U}^j_t,\boldsymbol{\lambda}^j_t\geq 0} \quad & \sum_{k=t}^{t+N-1} \mathbb{1}_{x_F}(x^j_{k|t}) + \sum_{i=0}^{j-1}\sum_{k=\delta^i_t}^{T^i} \lambda^i_k J^i_{k\to T^i}(x^i_k) \\
\text{s.t.} \quad & x^j_{k+1|t} = f(x^j_{k|t}, u^j_{k|t}), \forall k = t,\cdots,t+N-1 \\
& x^j_{k|t} \in \mathcal{X}, u^j_{k|t} \in \mathcal{U}, \forall k = t,\cdots,t+N-1 \\
& \sum_{i=0}^{j-1}\sum_{k=\delta^i_t}^{T^i} \lambda^i_k x^i_k = x^j_{t+N|t}, \\
& \sum_{i=0}^{j-1}\sum_{k=\delta^i_t}^{T^i} \lambda^i_k = 1 \\
& x^j_{t|t} = x^j_t
\end{aligned} \tag{5.15}$$

where $\mathbf{U}^j_t = [u^j_{t|t},\dots,u^j_{t+N-1|t}] \in \mathbb{R}^{d\times N}$ and the vector $\boldsymbol{\lambda}^j_t = [\lambda^0_0,\dots,\lambda^{j-1}_{T^{j-1}}] \in \mathbb{R}^{\Pi_{i=0}^{j-1}T^i}$ describes the terminal constraint set $\mathcal{CS}^{j-1}_{t+N}$ and terminal cost function $\bar{Q}^{j-1}_{t+N}(\cdot)$. Let the optimal solution to (5.12) at time t of the jth iteration be

$$\begin{aligned}
\mathbf{U}^{j,*}_t &= [u^{j,*}_{t|t},\dots,u^{j,*}_{t+N-1|t}] \\
\boldsymbol{\lambda}^{j,*}_t &= [\lambda^{j,*}_0,\dots,\lambda^{j,*}_{T^{h-1}}].
\end{aligned} \tag{5.16}$$

Then, we apply to the system (5.1) the first element of the optimal input sequence,

$$u^j_t = \bar{\pi}^{\text{LMPC},j}_t(x^j_t) = u^{j,*}_{t|t}. \tag{5.17}$$

Notice that the terminal constraints in (5.15) is enforced using nonlinear equality constraint, and therefore the computation burden is reduced with respect to the LMPC from Section 5.3.1. In the next section we will show that for a class on nonlinear system the relaxed LMPC (5.15) and (5.17) has the same safety and performance improvement properties of the LMPC presented in Section 5.3.1.

5.4 Properties

This section describes the properties of the proposed control strategies. We show that the LMPC guarantees constraint satisfaction at all time instants, convergence in finite time to x_F and iterative performance improvement. Furthermore, we demonstrate that the same properties are guaranteed when the relaxed LMPC is in closed-loop with a specific class on nonlinear systems or when a sufficient condition on the stored data and the system dynamics is satisfied.

5.4.1 Recursive Feasibility

We assume that a feasible trajectory which drives the system from the starting point x_S to the terminal state x_F is given, and we show that the controller recursively satisfies state and input constraints (5.2).

Assumption 3 *We are given the closed-loop trajectory and associated input sequence*

$$\mathbf{x}^0 = [x_0^0, \dots, \ x_{T^0}^0] \text{ and } \mathbf{u}^0 = [u_0^0, \dots, \ u_{T^0}^0],$$

which satisfy state and input constraints (5.2)*. Furthermore, we have that* $x_0^0 = x_S$ *and* $x_{T^0}^0 = x_F$*.*

Theorem 9 *Consider system* (5.1) *controlled by the LMPC* (5.12) *and* (5.14)*. Let* $\mathcal{SS}_t^j$ *be the time varying safe set at iteration* j *as defined in* (5.5)*. Let Assumption 3 hold and assume that* $x_0^j = x_S \ \forall j \geq 0$*, then at every iteration* $j \geq 1$ *the LMPC* (5.12) *and* (5.14) *is feasible for all* $t \geq 0$ *when* (5.14) *is applied to system* (5.1)*.*

Proof *By definition at time* $t = 0$ *we have that the state trajectory and associated input sequence,*

$$[x_0^{j-1}, \dots, x_N^{j-1}] \text{ and } [u_0^{j-1}, \dots, u_{N-1}^{j-1}], \tag{5.18}$$

satisfy input and state constraint and therefore the LMPC at time $t = 0$ *of the* j*th iteration is feasible.*
Assume that the LMPC (5.12) *and* (5.14) *is feasible at time* t*, let* (5.13) *be the optimal solution and* $x_{t+N|t}^{j,*} = x_{k^*}^{i^*}$*, where* $x_{k^*}^{i^*}$ *is defined in* (5.10)*. Then, we have that the following state trajectory and associated input sequence*

$$\begin{aligned} &[x_{t+1|t}^{j,*}, \dots, x_{t+N|t}^{j,*} = x_{k^*}^{i^*}, x_{k^*+1}^{i^*}] \\ &[u_{t+1|t}^{j,*}, \dots, u_{t+N-1|t}^{j,*}, u_{k^*}^{i^*}], \end{aligned} \tag{5.19}$$

satisfy input and state constraints (5.2) *and the LMPC at time* $t+1$ *of the* j*th iteration is feasible.*
Concluding, we have shown that the LMPC is feasible at time $t = 0$ *of the* j*th iteration. Furthermore, at each* j*th iteration we have that if the LMPC is feasible at time* t*, then the LMPC is feasible at time* $t+1$*. Therefore we conclude by induction that the LMPC* (5.12) *and* (5.14) *is feasible for all* $t \geq 0$ *and iteration* $j \geq 1$*.*

Next we consider a specific class on nonlinear systems which satisfies the following assumption.

Assumption 4 *Given any* P *states* $x_i \in \mathcal{X}$ *and input* $u_i \in \mathcal{U}$ *for* $i \in \{1, \dots, P\}$*, we have that* $\forall x \in Conv(x_1, \dots, x_P)$ *there exists* $u \in Conv(u_1, \dots, u_P)$ *such that*

$$f(x, u) \in Conv\big(f(x_1, u_1), \dots, f(x_P, u_P)\big)$$

where $f(\cdot, \cdot)$ *is defined in* (5.1)*.*

Finally, we show that if Assumption 4 is satisfied and the constraint sets in (5.2) are convex, then the relaxed LMPC (5.15) and (5.17) in closed-loop with system (5.1) guarantees recursive state and input constraint satisfaction.

Assumption 5 *The state and input constraint sets $\mathcal{X}$ and $\mathcal{U}$ in* (5.2) *are convex.*

Theorem 10 *Consider system* (5.1) *controlled by the relaxed LMPC* (5.15) *and* (5.17)*. Let $\mathcal{CS}_t^j$ be the convex safe set at iteration j as defined in* (5.7)*. Let Assumptions 3-5 hold and assume that $x_0^j = x_S \ \forall j \geq 0$, then at every iteration $j \geq 1$ the relaxed LMPC* (5.15) *and* (5.17) *is feasible for all $t \geq 0$ when* (5.17) *is applied to system* (5.1)*.*

Proof *We notice that by Assumption 4 it follows that $\forall x \in \mathcal{CS}_t^j$ there exists $u \in \mathcal{U}$ such that $f(x,u) \in \mathcal{CS}_{t+1}^j$. Therefore, if at time t of iteration j the relaxed LMPC* (5.15) *and* (5.17) *is feasible with optimal solution* (5.16)*, we have that there exists a state trajectory and related input sequence*

$$[x_{t+1|t}^{j,*}, \dots, x_{t+N|t}^{j,*}, f(x_{t+N|t}^{j,*}, u) \in \mathcal{CS}_{t+1}^j]$$
$$[u_{t+1|t}^{j,*}, \dots, u_{t+N-1|t}^{j,*}, u \in \mathcal{U}],$$

which satisfy state and input constraints (5.2) *and therefore the relaxed LMPC* (5.15) *and* (5.17) *is feasible at time $t+1$. Using this fact, the rest of the proof follows as for Theorem 9.*

5.4.2 Convergence and Performance Improvement

We show that the closed-loop system (5.1) and (5.14) converges in finite time to the terminal state x_F. Furthermore, the time T^j at which the closed-loop system converges to the terminal state x_F is non-increasing with the iteration index, i.e. $T^j \leq T^i, \forall i \in \{0, \dots, j-1\}$. In the following, we present a side result which will be used in the main theorem.

Proposition 2 *Consider system* (5.1) *controlled by the LMPC* (5.12) *and* (5.14)*. Assume that $\mathcal{SS}_t^{j-1} = x_F$ and $Q_t^{j-l} = 0$ for all $t \geq 0$. If at time t Problem* (5.12) *is feasible, then the closed-loop system* (5.1) *and* (5.14) *converges in at most $t+N$ time steps to x_F.*

Proof *Since $\mathcal{SS}_t^{j-1} = x_F$ is an invariant and $Q_t^{j-l} = 0$, we have that the LMPC* (5.12) *and* (5.14) *is feasible at all time instants and*

$$J_{t\to t+N}^{LMPC,j}(x_t^j) \geq h(x_t) + J_{t+1\to t+1+N}^{LMPC,j}(x_t^j). \tag{5.20}$$

Now, we assume that $x_i \neq x_F \forall i \in \{t, \dots, t+N-1\}$. Therefore by (5.20) *we have that at time $k = t+N-1$*

$$J_{k\to k+N}^{LMPC,j}(x_k^j) \leq J_{t\to t+N}^{LMPC,j}(x_t^j) - \sum_{i=t}^{k} h(x_i) = 1$$

which implies that $x_{k+1} = x_{t+N} = x_{t+N|t+N-1}^{j,*} = x_F$.

Theorem 11 *Consider system* (5.1) *controlled by the LMPC* (5.12) *and* (5.14). *Let* $\mathcal{SS}_t^j$ *be the time varying safe set at iteration* j *as defined in* (5.5). *Let Assumption 3 hold and assume that* $x_0^j = x_S \ \forall j \geq 0$, *then the time* T^j *at which the closed-loop system* (5.1) *and* (5.14) *converges to* x_F *is non-increasing with the iteration index,*

$$T^j \leq T^k, \ \forall k \in \{0, \dots, j-1\}.$$

Proof *By Theorem 9 we have that the LMPC* (5.12) *and* (5.14) *is feasible for all time* $t \geq 0$. *Denote*

$$T^{j-1,*} = \min_{k \in \{0,\dots,j-1\}} T^k$$

as the minimum to complete the task associated with the trajectories used to construct $\mathcal{SS}_{t+N}^{j-1}$. *By definitions* (5.5)-(5.6), *we have that at time* $\bar{t} = T^{j-1,*} - N \geq 0$

$$SS_{\bar{t}+N}^{j-1} = SS_{T^{j-1,*}}^{j-1} = x_F.$$

Therefore, by Proposition 2 the closed-loop system converges at most in $\bar{t} + N = T^{j-1,*}$ *steps. Finally, we notice that* $T^j = \bar{t} + N = T^{j-1,*} \leq T^k, \ \forall k \in \{0, \dots, j-1\}$.

Next, we show that if the relaxed LMPC (5.15) and (5.17) is in closed-loop with system (5.1) which satisfied Assumption 4, then T^j is non-increasing with the iteration index. The proof follows as in Theorem 11 exploiting the recursive feasibility of the relaxed LMPC (5.15) and (5.17) from Theorem 10.

Proposition 3 *Consider system* (5.1) *controlled by the LMPC* (5.15) *and* (5.17). *Assume that* $\mathcal{CS}_t^{j-1} = x_F$ *and* $\bar{Q}_t^{j-l} = 0$ *for all* $t \geq 0$. *If at time* t *Problem* (5.15) *is feasible, then the closed-loop system* (5.1) *and* (5.17) *converges in at most* $t + N$ *time steps to* x_F.

Proof *The proof follows as in Proposition 2 replacing the LMPC cost* $J_{t \to t+N}^{LMPC,j}(\cdot)$ *with the relaxed LMPC cost* $\bar{J}_{t \to t+N}^{LMPC,j}(\cdot)$.

Theorem 12 *Consider system* (5.1) *controlled by the LMPC* (5.15) *and* (5.17). *Let* $\mathcal{CS}_t^j$ *be the time varying safe set at iteration* j *as defined in* (5.7). *Let Assumptions 3-5 hold and assume that* $x_0^j = x_S \ \forall j \geq 0$, *then the time* T^j *at which the closed-loop system* (5.1) *and* (5.14) *converges to* x_F *is non-increasing with the iteration index,*

$$T^j \leq T^k, \ \forall k \in \{0, \dots, j-1\}.$$

Proof *By Theorem 10 we have that Problem* (5.15) *is feasible at all time* $t \geq 0$. *Therefore, the proof follows as for Theorem 11 using Proposition 3.*

5.4.3 Sufficient Condition for the Relaxed LMPC

In the previous sections we discussed the properties of the relaxed LMPC strategy in closed-loop with nonlinear systems which satisfy Assumption 4. Next, we show that the recursive constraint satisfaction and performance improvement properties properties still hold, if we replace Assumption 4 with the following assumption on the system dynamics and stored data.

Assumption 6 *Consider j stored feasible closed-loop trajectories $\mathbf{x}^j$ and associated input sequences $\mathbf{u}^j$. For all $x \in \mathbb{R}^n$ which can be expressed as convex combination of $n+1$ stored states, i.e.*

$$x \in Conv\Big(\bigcup_{\{t,i\}\in\mathcal{I}(x)} x_t^i\Big),$$

where the set $I(x) = \{\{t_0, i_0\}, \dots, \{t_n, i_n\}\}$ collects $n+1$ time and iteration indices associated with the stored states, we have that there exists $u \in \mathcal{U}$ such that

$$f(x,u) \in Conv\Big(\bigcup_{\{t,i\}\in\mathcal{I}(x)} f(x_t^i, u_t^i)\Big).$$

We underlined that the above assumption is hard to verify in general. In practice, Assumption 6 may be approximately checked using sampling strategies, as shown in the result section.

Finally, we state the following theorem which summaries the necessary conditions that guarantee recursive constraint satisfaction, convergence in finite time and iterative performance improvement for the relaxed LMPC in closed-loop with the nonlinear system (5.1).

Theorem 13 *Consider system* (5.1) *controlled by the relaxed LMPC* (5.15) *and* (5.17). *Let $\mathcal{CS}_t^j$ be the time varying convex safe set at iteration j as defined in* (5.7). *Let Assumptions 3, 5 and 6 hold and assume that $x_0^j = x_S \ \forall j \geq 0$. Then, the relaxed LMPC* (5.15) *and* (5.17) *satisfies state and input constraints* (5.2) *at all time. Furthermore, the time T^j at which the closed-loop system* (5.1) *and* (5.17) *converges to x_F is non-increasing with the iteration index,*

$$T^j \leq T^k, \ \forall k \in \{0, \dots, j-1\}.$$

Proof *We assume that at time t the relaxed LMPC* (5.15) *and* (5.17) *is feasible, let* (5.13) *be the optimal solution. As Assumption 6 holds, we have that there exists $u \in \mathcal{U}$ such that*

$$[x_{t+1|t}^{j,*}, \dots, x_{t+N|t}^{j,*}, f(x_{t+N|t}^{j,*}, u) \in \mathcal{CS}_{t+1}^j]$$
$$[u_{t+1|t}^{j,*}, \dots, u_{t+N-1|t}^{j,*}, u \in \mathcal{U}],$$

satisfy state and input constraints (5.2), *and therefore the relaxed LMPC* (5.15) *and* (5.17) *is feasible at time $t+1$. The rest of the proof follows as in Theorems 10 and 12.*

5.5 Data Reduction

In this section, we show that the proposed LMPC can be implemented using a subset of the time varying safe set from (5.5). In particular, we show we the controller may be implemented using the last l iterations and P data points per iteration.

5.5.1 Safe Subset

We define the time varying safe subset from iteration l to iteration j and for P data points

$$\mathcal{SS}^{j,l}_{t,P} = \bigcup_{i=l}^{j} \bigcup_{k=\delta^i_t}^{\delta^i_t+P} x^i_k, \tag{5.21}$$

where δ^i_t is defined in (5.6). Furthermore, in the above definition we set $x^i_k = x_F$ for all $k > T^i$ and $i < 0$. A representation of the time varying safe subset for a two-dimensional system is shown in Figure 5.3. Compare the safe subset $\mathcal{SS}^{j,l}_{t,P}$ with the safe set $\mathcal{SS}^j_t$ from (5.5). We notice that, $\mathcal{SS}^{j,l}_{t,P}$ is contained into $\mathcal{SS}^j_t$. Therefore, at time t the safe subset collects the stored states from which system (5.1) can reach the terminal state x_F in at most $(T^{j,*} - t)$ time steps. Finally, by definition, if a state x^i_t belongs to $\mathcal{SS}^{j,l}_{t,P}$, then there exists a feasible control action $u^i_t \in \mathcal{U}$ which keeps the evolution of the nonlinear system (5.1) into the time varying safe set at the next time step $t+1$, i.e. $f(x^i_t, u^i_t) \in \mathcal{SS}^{j,l}_{t+1,P}$. This property allows us to replace $\mathcal{SS}^{j,l}_{t,P}$ with $\mathcal{SS}^j_t$ in the design of the LMPC (5.14) and (5.12), without loosing the recursive constraint satisfaction property from Theorem 9.

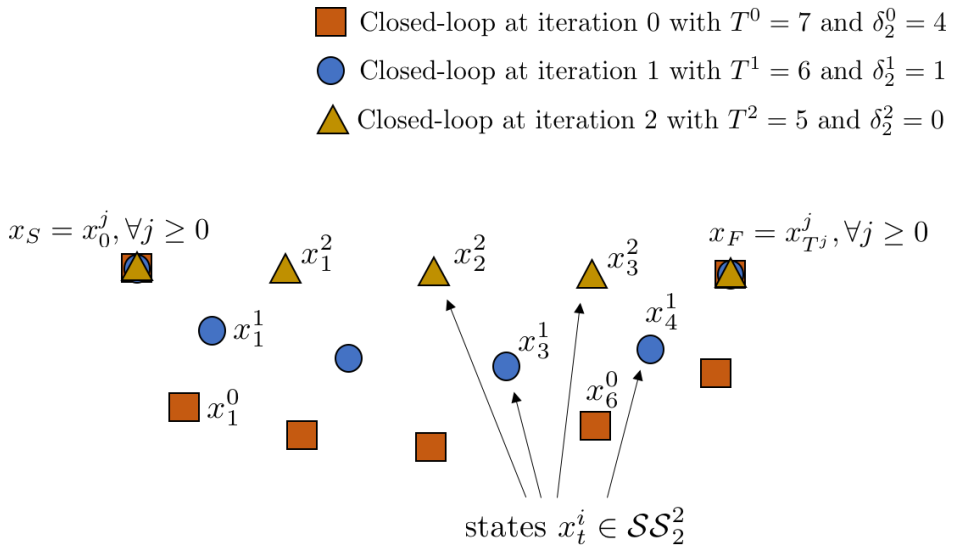


Figure 5.3: Representation of the time varying safe subset $\mathcal{SS}^{2,1}_{2,2}$. We notice that just a subset of the stored states are used to define $\mathcal{SS}^{2,1}_{2,2}$.

Finally, at each time t the we define the local convex safe subset as the convex hull of $\mathcal{SS}^{j,l}_{t,P}$ from (5.5),

$$\mathcal{CS}^{j,l}_{t,P} = \text{Conv}\big(\mathcal{SS}^{j,l}_{t,P}\big). \tag{5.22}$$

We underline that relaxed LMPC from Section 5.3.2 may be implemented replacing the the convex safe set (5.7) with the convex safe subset (5.22).

5.5.2 Q-function

In the section, we construct the Q-function which assigns the cost-to-go to the states contained into the time varying safe subset from (5.21). In particular, we introduce the function $Q^{j,l}_{t,P}(\cdot)$, defined over the safe subset $\mathcal{SS}^{j,l}_{t,P}$, as

$$\begin{aligned} Q^{j,l}_{t,P}(x) = \min_{\substack{i\in\{l,\dots,j\}\\ t\in\{\delta^i_t,\dots,\delta^i_t+P\}}} \quad & J^i_{t\to T^i}(x^i_t) \\ \text{s.t.} \quad & x = x^i_t \in \mathcal{SS}^{j,l}_{t,P}. \end{aligned} \tag{5.23}$$

Compare the above function $Q^{j,l}_{t,P}$ with Q^j_t from (5.9). We notice that, the domain of $Q^{j,l}_{t,P}$ is the safe subset $\mathcal{SS}^{j,l}_{t,P}$ and the domain of the Q^j_t is the safe set $\mathcal{SS}^j_t \supseteq \mathcal{SS}^{j,l}_{t,P}$. Moreover, we have that

$$\forall x \in \mathcal{SS}^{j,l}_{t,P}, Q^{j,l}_{t,P}(x) = Q^j_t(x).$$

Therefore, if we replace Q^j_t with $Q^{j,l}_{t,P}$ in the design of the LMPC policy (5.14), then the finite time convergence and performance improvements properties still hold.

Furthermore, we define the convex Q-function $\bar{Q}^{j,l}_{t,P}$ from iteration l to iteration j and P data points as

$$\begin{aligned} \bar{Q}^{j,l}_{t,P}(x) = \min_{[\lambda^0_{\delta^i_t},\dots,\lambda^j_{\delta^i_t+P}]\geq 0} \quad & \sum_{i=0}^{j}\sum_{k=\delta^i_t}^{\delta^i_t+P} \lambda^i_k J^i_{k\to\delta^i_t+P}(x^i_k) \\ \text{s.t.} \quad & \sum_{i=0}^{j}\sum_{k=\delta^i_t}^{\delta^i_t+P} \lambda^i_k x^i_k = x \\ & \sum_{i=0}^{j}\sum_{k=\delta^i_t}^{\delta^i_t+P} \lambda^i_k = 1. \end{aligned} \tag{5.24}$$

where δ^i_t is defined in (5.6). The above convex Q-function $\bar{Q}^j_t(\cdot)$ is simply a piecewise-affine interpolation of Q-function from (5.23) over the convex safe subset, as shown in Figure 5.4. We underline that $\bar{Q}^{j,l}_{t,P}$ can be used in the relaxed LMPC design instead of $\bar{Q}^j_t$.

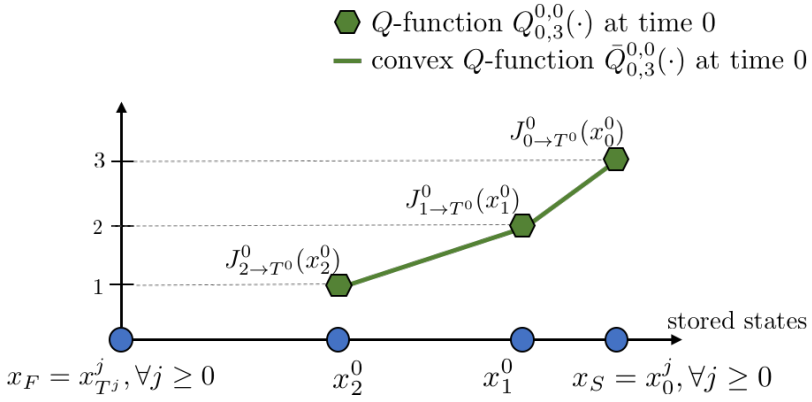


Figure 5.4: Notice that the Q-function $Q^{0,0}_{0,3}(\cdot)$ is defined over a set of discrete data points, whereas the convex Q-function $\bar{Q}^{0,0}_{0,3}(\cdot)$ is defined over the convex safe set.

5.6 Examples

We test the proposed strategy on 3 time optimal control problems. In first example, the LMPC is used to drive a dubins car from the staring point x_S to the terminal point x_F while avoiding an obstacle. In the second example, we control a nonlinear double integrator system, which satisfies Assumption 4. Finally, the third example is a dubins car racing problem, which we solved using the relaxed LMPC after checking Assumption 6 via sampling. The code for these examples is available at `https://github.com/urosolia/LMPC` in the NonlinearLMPC folder.

5.6.1 Minimum time obstacle avoidance

We use the LMPC policy synthesized with the mixed integer approach from Section 5.3.1 on the minimum time obstacle avoidance optimal control problem from Section 4,

$$
\begin{aligned}
\min_{\substack{T,a_0,\dots,a_{T-1}\\ \theta_0,\dots,\theta_{T-1}}} \quad & \sum_{t=0}^{T-1} 1 \\
\text{s.t.} \quad & \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + v_t\cos(\theta_t) \\ y_t + v_t\sin(\theta_t) \\ v_t + a_t \end{bmatrix}, \forall t \geq 0 \\
& \frac{(x_t - x_{\text{obs}})^2}{a_x^2} + \frac{(y_t - y_{\text{obs}})^2}{a_y^2} \geq 1, \forall t \geq 0 \\
& \begin{bmatrix} -\pi/2 \\ -1 \end{bmatrix} \leq \begin{bmatrix} \theta_t \\ a_t \end{bmatrix} \leq \begin{bmatrix} \pi/2 \\ 1 \end{bmatrix}, \forall t \geq 0 \\
& x_T = x_F = [54, 0, 0]^T, \\
& x_0 = x_S = [0, 0, 0]^T.
\end{aligned}
$$

The goal of the above minimum time optimal control problem is to steer the dubins car from the starting state x_S to the terminal point x_F, while satisfying input saturation constraints and avoiding an obstacle. The obstacle is represented by an ellipse centered at $(x_{\text{obs}}, y_{\text{obs}}) = (27, -1)$ with semi-axis $(a_x, a_y) = (8, 6)$. At iteration 0, we compute a first feasible trajectory using a brute force algorithms and we use the closed-loop data to initialize the LMPC (5.12) and (5.14) with $N = 6$.

We compare the performance of the LMPC from [79] and of the LMPC policies (5.14) synthesized using different number of data points $P = \{8, 10, 40\}$ and iterations $i = \{1, 2, 3\}$, as described in Section 5.5 (i.e. basically in definition (5.21) we set $l = j - 1 - i$). Figure 5.5 shows the time T^j at which the closed-loop system converged to the terminal state x_F at each iteration index. We notice that all LMPC policies converge to a steady state behavior which steers the system from x_S to x_F in 16 time steps. Furthermore, Figure 5.5 shows that the number of iterations needed to reach convergence is proportional to the amount of data used to synthesize the LMPC policy.

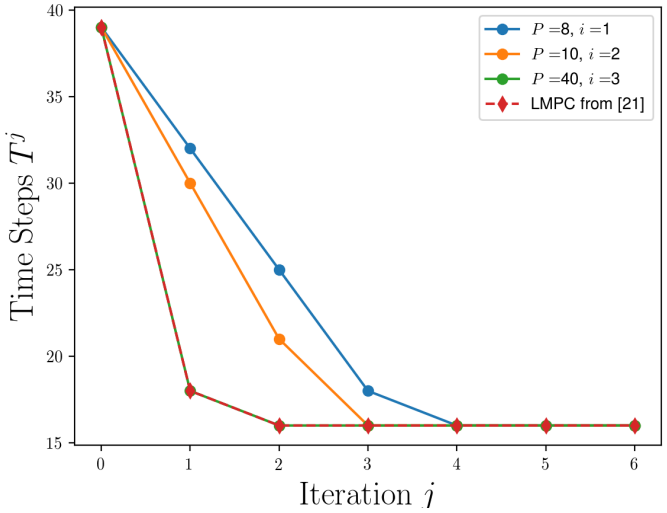


Figure 5.5: Time steps T^j to reach x_F as a function of the iteration index. We notice that as more data points are used in the synthesis process, the number of iterations needed to reach a steady state behavior decreases.

Figure 5.6 shows that the computational time increases as more data points P are used in the control design. Therefore, there is a trade-off between the computational burden and the performance improvement at each iteration from Figure 5.5. It is interesting to notice that the computational cost associated with the proposed time varying LMPC strategy converges to a steady state value. On the other hand, the computation cost associate with the LMPC strategy from [79] increases at each iteration. Therefore, we confirm that the proposed time varying LMPC (5.12) and (5.14) allows us to reduce the computational cost while achieving the same closed-loop performance.

Finally, we analyse the closed-loop trajectories associated with the LMPC policy (5.14)

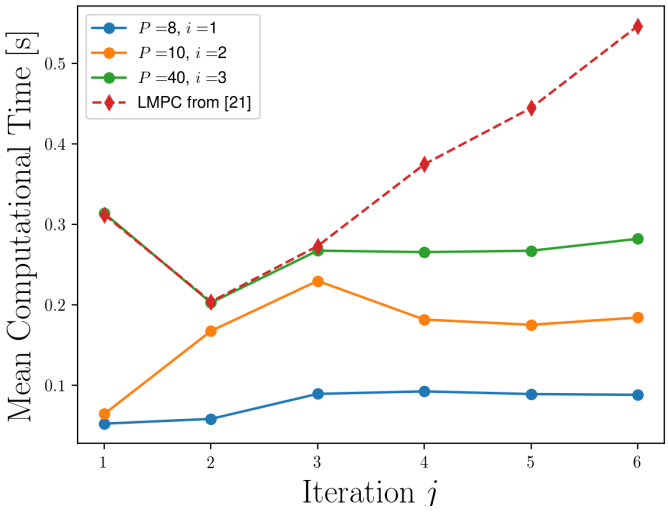


Figure 5.6: Computational cost associated with the LMPC policy at each time t as function of the iteration index. We notice that as more data points are used in the synthesis process, the computational cost increases.

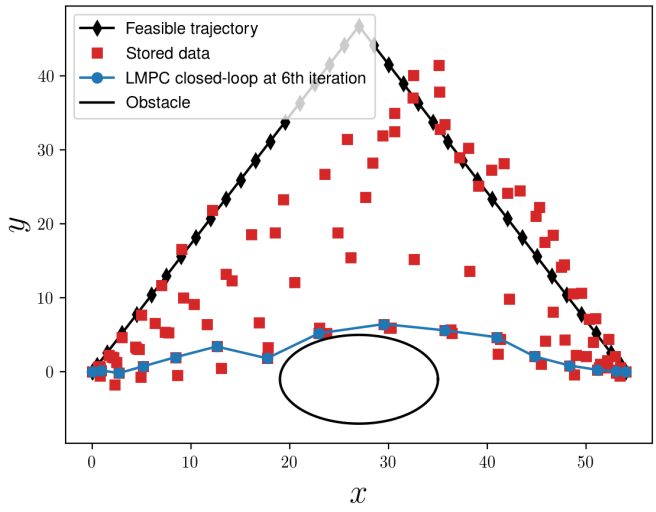


Figure 5.7: First feasible trajectory, stored data points and closed-loop trajectory at the 6th iteration. We notice that the LMPC is able to avoid the obstacle at each iteration.

synthesized with $P = 8$ data points and $i = 1$ iteration. Figure 5.7 shows the first feasible trajectory, the stored data points and the closed-loop trajectory at convergence. We confirm that the LMPC is able to explore the state space while avoiding the obstacle and steering the system from the starting state x_S to the terminal state x_F. Furthermore, we notice that the LMPC accelerates during the first part of the task, and then it decelerates to reach the terminal state with zero velocity, as shown in Figure 5.8.

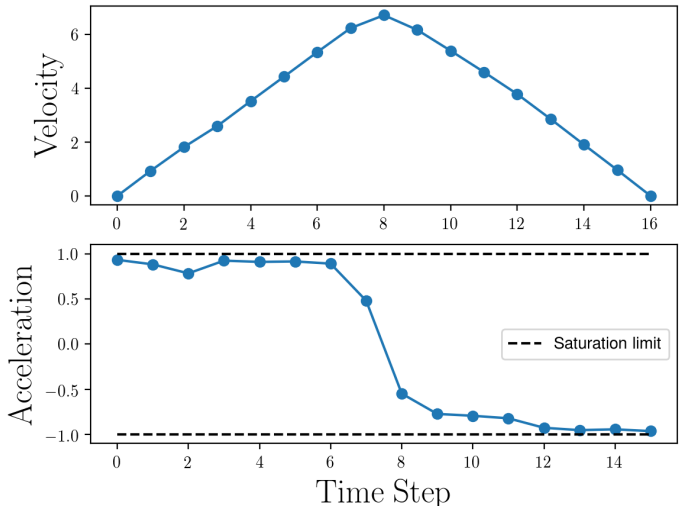


Figure 5.8: Acceleration and speed profile at convergence. We notice that the controller accelerates for the first 8 time steps and afterwards it decelerates to reach the terminal state goal state with zero velocity.

5.6.2 Nonlinear Double Integrator

In this section, we test the relaxed LMPC (5.15) and (5.17) on the following nonlinear double integrator problem

$$\begin{aligned} \min_{T,a_0,\dots,a_{T-1}} \quad & \sum_{t=0}^{T-1} 1 \\ \text{s.t.} \quad & \begin{bmatrix} x_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + v_t dt \\ v_t + \left(1 - \frac{v_t^2}{v_{\max}^2}\right) a_t dt \end{bmatrix}, \forall t \geq 0 \\ & 0 \leq v_t \leq v_{\max}, \forall t \geq 0 \\ & -1 \leq a_t \leq 1, \forall t \geq 0 \\ & x_T = x_F = [0, 0]^T, \\ & x_0 = x_S = [-10, 0]^T, \end{aligned} \tag{5.25}$$

where the state of the system are the velocity v_t and the position x_t. The control action is the acceleration a_t which is scaled by the concave function $g(v_t) = \left(1 - v_t^2/v_{\max}^2\right)$. In Section 5.7.1 of the Appendix we show that the above nonlinear double integrator satisfies Assumption 4. We used an handcrafted policy to perform the first feasible trajectory used to initialize the relaxed LMPC policies synthesized with $N = 4$. Furthermore, we implemented the strategy from Section 5.5 using $P = \{12, 25, 50, 200\}$ data points and $i = \{1, 3, 4, 10\}$ iterations.

Figures 5.9 shows the number of iterations needed to reach convergence. We notice that as more data points P are used in the policy synthesis process, the closed-loop system

convergence faster in the iteration domain to a trajectory which performs the task in 14 time steps.

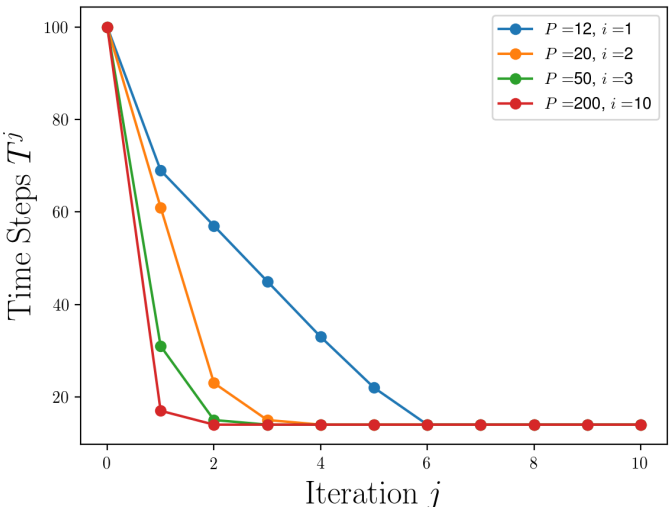


Figure 5.9: Time steps T^j to reach x_F as a function of the iteration index. We notice that, also in this example, as more data points are used in the synthesis process, the number of iterations needed to reach a steady state behavior decreases.

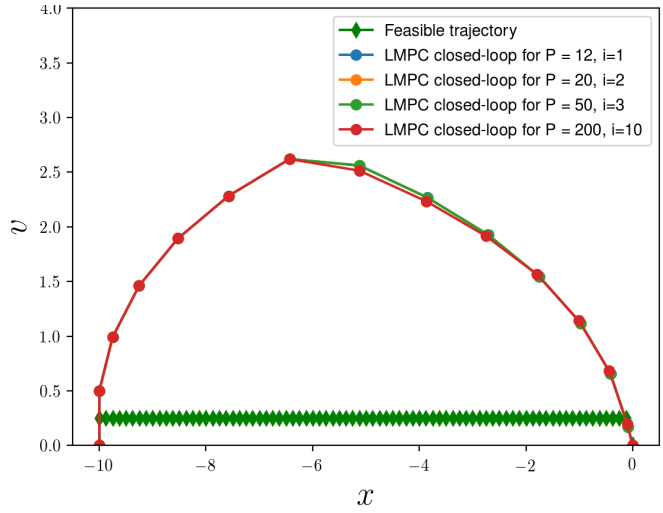


Figure 5.10: First feasible trajectory and closed-loop trajectories at the 10th iteration. We notice that all LMPC policies converged to as similar behavior.

Finally, Figures 5.10 and 5.11 show the steady-state closed-loop trajectories and the associated input sequences for all tested policies. We notice that after few iterations of the control task, the closed-loop systems converged to a similar behavior. In particular, the controller saturates the acceleration and deceleration constraints, as we would expect from

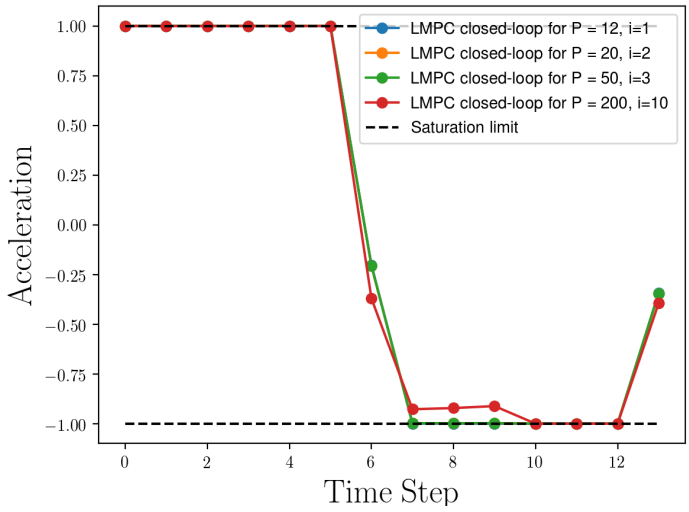


Figure 5.11: Acceleration inputs associated with the closed-loop trajectories at the 10th iteration. We notice that the controller saturated the acceleration constraints.

the optimal solution to a time optimal control problem (Fig. 5.11). It is interesting to notice that accelerating the nonlinear double integrator to a peak speed requires more control effort than slowing down the system to zero speed. Therefore, the controller accelerates for the first 6 time steps and then it decelerates for the last 8 time steps to reach the terminal state with zero velocity.

5.6.3 Minimum Time Dubins Car Racing

We test the relaxed LMPC (5.15) and (5.17) on a minimum time racing problem. The goal of the controller is to drive the dubins car on a curve of constant radius $R = 10$ from the staring point x_S to the finish line. More formally, we would like to solve the following minimum time optimal control problem

$$\begin{aligned}
\min_{\substack{T,a_0,\dots,a_{T-1}\\ \theta_0,\dots,\theta_{T-1}}} \quad & \sum_{t=0}^{T-1} 1 \\
\text{s.t.} \quad & \begin{bmatrix} s_{t+1} \\ e_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} s_t + \frac{v_t \cos(\theta_t - \gamma(s_t))}{1 - e_t/R} dt \\ e_t + v_t \sin(\theta_t - \gamma(s_t)) dt \\ v_t + a_t dt \end{bmatrix}, \forall t \geq 0 \\
& \begin{bmatrix} -2 \\ -1 \end{bmatrix} \leq \begin{bmatrix} \theta_t \\ a_t \end{bmatrix} \leq \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \forall t \geq 0 \\
& e_{\min} \leq e_t \leq e_{\max}, \forall t \geq 0 \\
& x_T \in \mathcal{X}_F, \\
& x_0 = x_S = [0, 0, 0]^T,
\end{aligned} \tag{5.26}$$

where $\gamma(s_t)$ is the angle of the tangent vector to the centerline of the road at the curvilinear abscissa s_t, the discretization time $dt = 0.5$ and the lane half width $e_{\max} = -e_{\min} = 2.0$. The control actions are the heading angle θ_t and the acceleration command a_t. The system is represented in the curvilinear abscissa reference frame where the state s_t, e_t and v_t are the distance travelled along the centerline, the lateral distance from the center of the lane and the velocity, respectively. Notice that in the curvilinear abscissa reference frame the lane boundaries are represented by convex constraints on the state e_t, and therefore Assumption (5) is satisfied. The finish line is described by the following terminal set

$$\mathcal{X}_F = \left\{ x \in \mathbb{R}^3 \middle| \begin{bmatrix} 18.19 \\ -e_{\min} \\ 0 \end{bmatrix} \leq x \leq \begin{bmatrix} 18.69 \\ e_{\min} \\ 0 \end{bmatrix} \right\}. \tag{5.27}$$

As mentioned in Remark 7, in order to implement the LMPC to steer the system to terminal set instead of a terminal point, we replaced $x^i_{T^i} = x_F$ with the vertices of $\mathcal{X}_F$ in definitions (5.7) and (5.11).

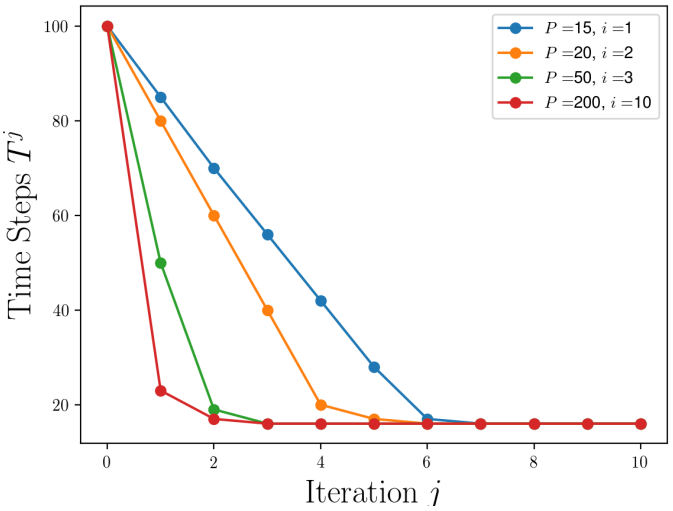


Figure 5.12: Time steps T^j to reach x_F as a function of the iteration index. We notice that as more points P and iterations i are used to synthesize the relaxed LMPC policy, the closed-loop system converges faster to a steady state behavior.

In order to compute the first feasible trajectory needed to initialize the LMPC, we set $\theta^0_t = \gamma(s^0_t)$ and we controlled the acceleration to steer the dubins car from x_S to the terminal set $\mathcal{X}_F$. Notice that for $\theta^0_t = \gamma(s^0_t)$ the system is linear and consequently Assumption 6 is satisfied for iteration $j = 0$. However, for $j > 0$ it is hard to verify analytically if Assumption 6 holds, therefore we used a sampling strategy to approximately check this condition, as shown in the Appendix 5.7.2.

We test different LMPC policies synthesised with $N = 4$ and using the strategy described in Section 5.5 for $P = \{15, 25, 50, 200\}$ data points and $i = \{1, 3, 4, 10\}$ iterations. Figure 5.12

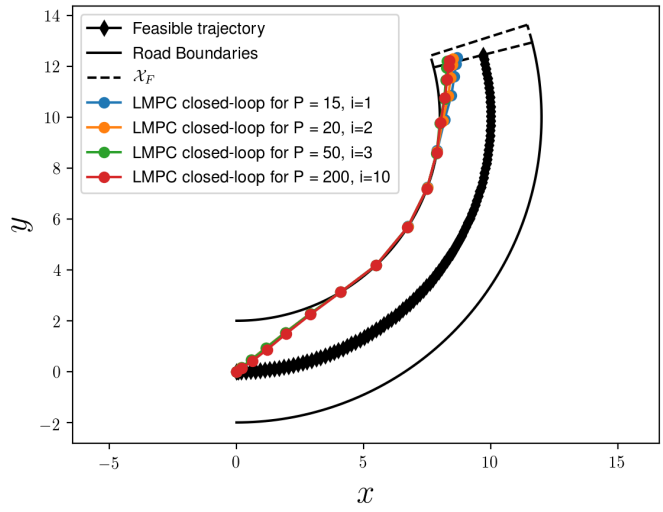


Figure 5.13: Comparison between the first feasible trajectory used to initialize the LMPC and the steady state LMPC closed-loop trajectories at convergence.

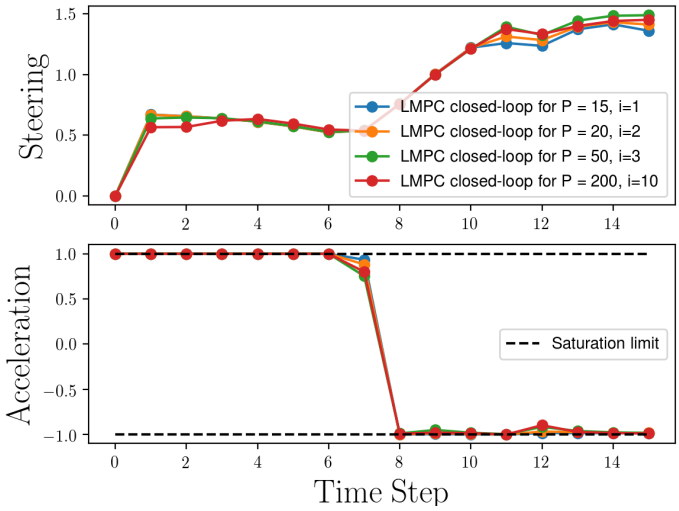


Figure 5.14: Comparison of the steady state inputs associated with the relaxed LMPC policies. We notice that the acceleration and deceleration is saturated, as we expect from the optimal solution to a minimum time optimal control problem.

shows time steps T^j needed to reach the terminal set (5.27). We notice that after few iterations all LMPC policies converged to a steady state behavior which steers the system to the goal set in 16 time steps. Also in this example, convergence is reached faster as more data points are used in the LMPC synthesis process.

Finally, Figures 5.13 and 5.14 show that closed-loop trajectories and associated input sequence at converged. In order to minimize the travel time, the LMPC cuts the curve and it drives the system to a state in the terminal set which is close to the road boundaries.

Furthermore, we notice that the controller saturates the acceleration and deceleration constraints, as we expect from an optimal solution to a minimum time optimal control problem.

5.7 Appendix

5.7.1 Nonlinear Double Integrator

In this section, we show that the following nonlinear double integrator

$$z_{k+1} = \begin{bmatrix} x_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_k dt \\ v_k + g(v_k) a_k dt \end{bmatrix} = f_n(z_k, a_k)$$

for $g(v_k) = (1 - v_k^2/v_{\max}^2)$ satisfies Assumption 4. First we notice that given P states $x_i \in \mathcal{X}$ and inputs $u_i \in \mathcal{U}$ for $i \in \{1, \dots, P\}$ and a set of multiplies $[\lambda_0, \dots, \lambda_P] \geq 0$

$$x = \sum_{k=1}^{P} \lambda_k x_k \text{ and } \sum_{k=1}^{P} \lambda_k = 1,$$

we have that

$$\sum_{k=0}^{P} \lambda_k f_n(z_k, a_k) = \sum_{k=0}^{P} f_n(\lambda_k z_k, a)$$

where

$$a = \frac{\sum_{k=0}^{P} \lambda_k g(v_k) a_k}{g\big(\sum_{k=0}^{P} \lambda_k v_k\big)}.$$

Finally, by concavity of $g(v_k) \geq 0$ for all $z_k = [x_k, v_k]^T \in \mathcal{X}$ we have that

$$\min_{k=1,\dots,P} a_k \leq \frac{\sum_{k=0}^{P} \lambda_k g(v_k) a_k}{g\big(\sum_{k=0}^{P} \lambda_k v_k\big)} \leq \max_{k=1,\dots,P} a_k$$

and therefore Assumption 4 is satisfied.

5.7.2 Dubins Car

We used a sampling strategy to check if Assumption 6 is approximately satisfied. Before running the $(j+1)$th iteration of the relaxed LMPC, we randomly sample

$$x^{(l)} \in \text{Conv}\Big(\bigcup_{\{t,i\} \in \mathcal{I}(x^{(l)})} x_t^i\Big) \text{ for } l \in \{0, \dots, 10^5\},$$

where $\mathcal{I}(x^{(l)})$ is defined in Assumption 6. Afterwards, we checked if $\exists u \in \mathcal{U}$ such that

$$f(x^{(l)}, u) \in \text{Conv}\Big(\bigcup_{\{t,i\} \in \mathcal{I}(x^{(l)})} f(x_t^i, u_t^i)\Big).$$

For all tested data points and iterations Assumption 6 was satisfied. Notice that as we used a subset of the stored data to construct (5.7) and (5.9), we checked Assumption 6 for the stored closed-loop trajectory performed at iteration j. Finally, for $j = \{3, 5, 10\}$ Figures 5.15, 5.16 and 5.17 show the randomly generated states where we have verified that Assumption 6 holds.

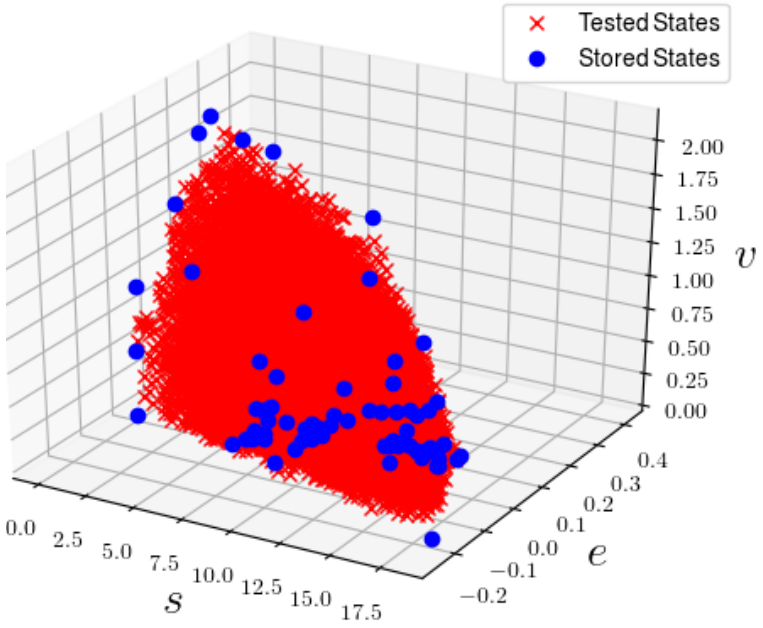


Figure 5.15: Randomly sampled states used to check that Assumption 6 is approximately satisfied.

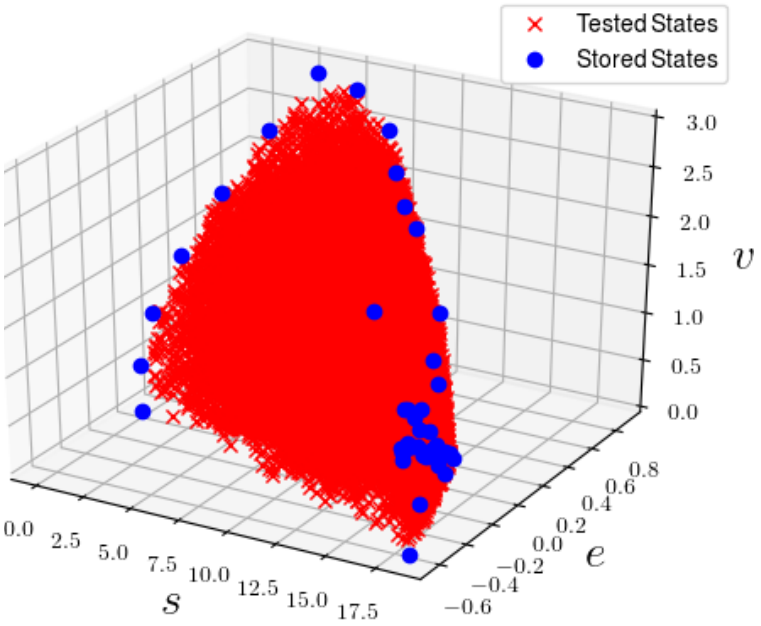


Figure 5.16: Randomly sampled states used to check that Assumption 6 is approximately satisfied.

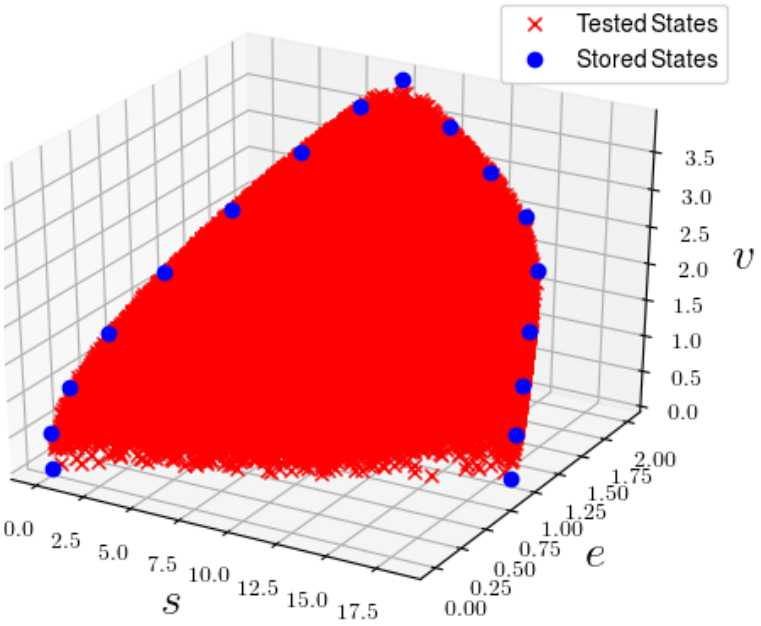


Figure 5.17: Randomly sampled states used to check that Assumption 6 is approximately satisfied.

Chapter 6

Learning Model Predictive Control for Uncertain Systems

In this chapter, we describe how to design LMPC policies for uncertain systems. First, we iteratively construct *robust safe sets* and *robust Q-functions*. Afterwards we present the LMPC synthesis process and we propose an approximation strategy that leverages closed-loop historical data to reduce the computational burden. Finally, we test the controller on a constraint LQR problem and on a parking example.

In order to better motivate this chapter, we describe the challenges associated with the computation of safe sets using historical data of uncertain systems. First, we recall how historical data can be used to compute invariant sets for deterministic systems. Consider the discrete time linear system

$$\bar{x}^j_{t+1} = A\bar{x}^j_t + B\pi^j(\bar{x}^j_t)$$

where $\pi^j(\cdot)$ is a feedback policy known only along the jth stored closed-loop trajectory $\bar{\mathbf{x}}^j = [\bar{x}^j_0, \dots, \bar{x}^j_t, \dots]$. Assume that $\pi^j(\cdot)$ is able to execute the desired task safely. At any iteration $i > j$ and time $k \geq 0$, if the system state x^i_k equals a state x^j_t which has been visited at the jth iteration, then the feedback policy $\pi^j(\cdot)$ will drive the system along the jth trajectory. This obvious fact is a consequence of the system being deterministic. More importantly, if the policy $\pi^j(\cdot)$ brings the system to an equilibrium point, then the convex hull of visited states is a control invariant set. Therefore, invariant sets for deterministic systems can be easily built from data.

In contrast, when dealing with *uncertain systems*, the set of visited states is not an invariant set. In fact, consider the discrete time uncertain system

$$x^j_{t+1} = Ax^j_t + B\pi^j(x^j_t) + w^j_t$$

where the random disturbance w^j_t belongs to the set $\mathcal{W}$ and the jth stored trajectory is $\mathbf{x}^j = [x^j_0, \dots, x^j_t, \dots]$. Assume that $\pi^j(\cdot)$ is able to execute the desired task safely at iteration j.

We notice that the stored trajectory $\mathbf{x}^j$ is associated with a specific disturbance realization $[w_0^j, \dots, w_t^j, \dots]$. For this reason, at any iteration $i > j$ and time $k \geq 0$, if the system state x_k^i equals a state x_t^j that has been visited, applying the feedback policy $\pi^j(\cdot)$ may drive the system to a state neither stored nor safe, due to a potentially different disturbance realization $[w_0^{j+1}, \dots, w_t^{j+1}, \dots]$. In conclusion, the set of visited states cannot be naively exploited to compute invariant sets.

For the above reason, one cannot simply use the deterministic Learning MPC approach presented in Chapter 4 for uncertain systems. In this chapter, we present a strategy to construct robust invariant sets for linear uncertain systems. Then, we use these sets in an iterative learning predictive control schema. Finally, we propose a synthesis procedure which leverages roll-outs of the closed-loop system to approximate the terminal cost and constraint.

6.1 Problem Formulation

We consider the following linear time invariant system

$$x_{t+1}^j = Ax_t^j + Bu_t^j + w_t^j \tag{6.1}$$

where at time t of the jth iteration the disturbance $w_t^j \in \mathcal{W}$, the state $x_t \in \mathbb{R}^n$ and input $u_t^j \in \mathbb{R}^d$. Furthermore, the system is subject to the following convex polytopic state and input constraints, for all $t \geq 0$

$$x_t^j \in \mathcal{X} \text{ and } \pi^j(x_t^j) \in \mathcal{U}.$$

At each jth iteration, we define the worst-case iteration cost associated with the control policy $\pi^j(\cdot)$, as the solution to the following Bellman recursion

$$J_{\pi^j}^j(x_0^j) = \max_{w \in \mathcal{W}} [h(x_0^j, u_0^j) + J_{\pi^j}^j(Ax_0^j + B\pi^j(x_0^j) + w)]. \tag{6.2}$$

The goal of the control design is to solve the following infinite time robust optimal control problem,

$$\begin{aligned} J_{0\to\infty}^{j,*}(x_S^j) = \min_{\pi^j(\cdot)} \quad & J_{\pi^j}^j(x_0^j) \\ & x_{k+1}^j = Ax_t^j + B\pi^j(x_t^j) + w_t^j, \\ & x_k^j \in \mathcal{X}, u_k^j \in \mathcal{U}, \\ & u_t^j = \pi^j(x_t^j), \forall t \in \{0, 1, \dots\} \\ & x_0^j = x_S^j, \\ & \forall t \in \{0, 1, \dots\}, \forall w_t^j \in \mathcal{W} \end{aligned} \tag{6.3}$$

In the following, we propose to solve the above Problem (6.3) iteratively. At each iteration j, we design a feedback policy

$$u_t^j = \pi^j(\cdot) : \mathcal{F}^j \subseteq \mathcal{X} \to \mathcal{U} \tag{6.4}$$

which guarantees *i*) *convergence* of the closed-loop system (6.1) and (6.4) to a neighborhood of the origin $\mathcal{O}$, *ii*) *safety*: state and input constraints are robustly satisfied, *iii*) *performance improvement*: if the controller performs the same task repeatedly (i.e. $x_0^j = x_0^{j+1}$), then the worst-case iteration cost (6.2) is non-decreasing (i.e. $J^{j+1}_{\pi^{j+1}}(x_0^{j+1}) \leq J^j_{\pi^j}(x_0^j)$), and *iv) exploration*: the domain of the policy (6.4) is not shrinking with the iteration index (i.e. $\mathcal{F}^i \subseteq \mathcal{F}^j, \forall j \geq i$).

Throughout this chapter we use the standard function classes $\mathcal{K}$, $\mathcal{K}_\infty$ and $\mathcal{KL}$ notation (see [22]) and we define the distance from a point $x \in \mathbb{R}^n$ to a set $\mathcal{O} \subseteq \mathbb{R}^n$ as

$$|x|_{\mathcal{O}} = \inf_{d \in \mathcal{O}} ||x - d||_1.$$

Furthermore, we make the following assumptions.

Assumption 7 *The set $\mathcal{O} \subset \mathbb{R}^n$ is a robust positive invariant set for the autonomous system $x_{t+1} = (A + BK)x_t + w_t$ and $w_t \in \mathcal{W}$,*

$$\forall x \in \mathcal{O} \rightarrow (A + BK)x_t + w_t \in \mathcal{O}, \forall w_t \in \mathcal{W}.$$

Assumption 8 *The continuous stage cost $h(\cdot, \cdot)$ is jointly convex in its arguments. Furthermore, we assume that $\forall x \in \mathbb{R}^n, \forall u \in \mathbb{R}^d$*

$$\alpha_x^l(|x|_{\mathcal{O}}) \leq h(x, 0) \leq \alpha_x^u(|x|_{\mathcal{O}}) \text{ and } \alpha_u^l(|u|_{K\mathcal{O}}) \leq h(0, u) \leq \alpha_x^u(|u|_{K\mathcal{O}}),$$

where $\alpha_x^u, \alpha_x^l, \alpha_u^u$ and $\alpha_u^l \in \mathcal{K}_\infty$.

6.2 Safe Set

In this section, we show how to iteratively construct a set of states from which the control task can be safely executed using the control policy (6.4). These safe sets will be used in the controller design to guarantee robust recursive constraint satisfaction.

First, we introduce the concise notation for the robust reachable set for the closed-loop system (6.1) and (6.4),

$$\mathcal{R}_{t+1}(x_0^j) = \left\{ x_{t+1} \in \mathcal{X} \middle| \begin{array}{c} \exists w_t \in \mathcal{W}, x_t \in \mathcal{R}_t(x_0^j), \\ x_{t+1} = Ax_t + B\pi^j(x) + w \end{array} \right\} \tag{6.5}$$

with $\mathcal{R}_0(x_0^j) = x_0^j$. The robust reachable set $\mathcal{R}_t(x_0^j)$ collects that states which may be reached by the closed-loop system (6.1) and (6.4) in t time steps.This property allows us to define the safe set at the jth iteration,

$$\mathcal{SS}^j = \left\{ \bigcup_{t=0}^{T^j} \mathcal{R}_t(x_0^j) \right\} \bigcup \mathcal{O}, \tag{6.6}$$

for a control task of T^j time steps. Notice that the safe set contains the state evolution of the closed-loop system (6.1) and (6.4) from the initial state x_0^j until completion of the control task. Furthermore, we underline that robust reachable sets (6.5) are computed propagating the vertices of the disturbance through the system dynamics. Therefore, the computational complexity of constructing the safe set $\mathcal{SS}^j$ explodes with the length of the control task T^j. For this reason in Section 6.5.1, we will show how to approximate the safe set $\mathcal{SS}^j$ using historical data from the closed-loop system (6.1) and (6.4).

Finally, we define the convex safe set $\mathcal{CS}^j$ as the convex hull of the safe sets $\mathcal{SS}^k$ for iterations $k \in \{0, \dots, j\}$,

$$\mathcal{CS}^j = \text{conv}\left(\bigcup_{k=0}^{j} \mathcal{SS}^k\right). \tag{6.7}$$

Figure 6.1 shows the robust convex safe set at iteration $j = 1$. Notice that, if the control policies $\pi^k(\cdot)$ for $k \in \{0, \dots, j\}$ safely steer the system to the neighborhood of the origin $\mathcal{O}$. Then, $\mathcal{CS}^j$ is a robust control invariant set as stated by the following proposition.

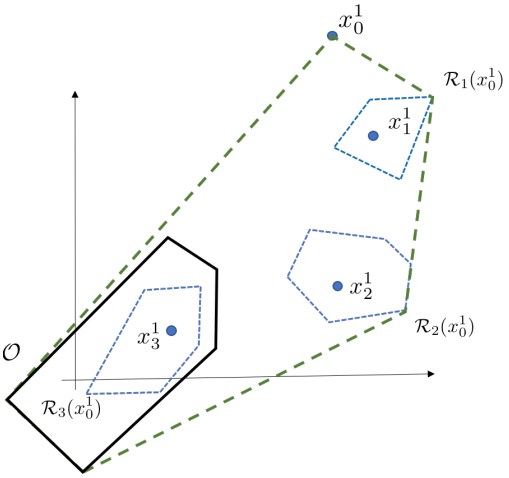


Figure 6.1: Representation of the robust convex safe set $\mathcal{CS}^1$ (dashed green line) at iteration $j = 1$. The figure reports also the N-steps robust reachable sets $\mathcal{R}_t(x_0^1)$ (dashed blue line) and the robust invariant set $\mathcal{O}$ (solid black line).

Proposition 4 *For $j \geq 1$, let $\pi^j(\cdot) : \mathcal{F}^j \to \mathcal{U}$ be a control policy defined over $\mathcal{F}^j \subseteq \mathcal{X}$. Consider system (6.1) in closed-loop with $\pi^j(\cdot)$ and assume that $\forall x_0^j \in \mathcal{F}^j$ we have $x_t^j \in \mathcal{X}$ and $x_{T^j}^j \in \mathcal{O}$ $\forall w_t \in \mathcal{W}, t \geq 0$. Then, the convex safe set $\mathcal{CS}^j$ is a robust control invariant set for system (6.1), i.e.*

$$\forall x \in \mathcal{CS}^j \to Ax + B\pi^j(x) + w \in \mathcal{CS}^j, \ \forall w \in \mathcal{W}.$$

Proof *By the assumptions on $\pi^k(\cdot)$ for $k \in \{0, \dots, j\}$ and definition (6.6), we have that $\mathcal{SS}^k$ is a robust control invariant set for $k \in \{0, \dots, j\}$. Therefore, by linearity of system (6.1) it follows that $\mathcal{CS}^j$ is a robust control invariant set.* ■

6.3 Q-function

The convex safe set from the previous section represents a subset of states from which the control task may be safely completed. Next, we define the value function $Q^j(\cdot) : \mathcal{CS}^j \to \mathbb{R}$, which approximates the cost-to-go from any state contained into the convex safe set. This function will allow us to guarantee worst-case performance improvement at each iteration.

Recall that the Bellman recursion (6.2) for the control policy $\pi^j(\cdot)$

$$J^j_{\pi^j}(x) = \max_{w \in \mathcal{W}} [h(x, \pi^j(x)) + J^j_{\pi^j}(Ax + B\pi^j(x) + w)] \tag{6.8}$$

represents the worst-case cost-to-go from any point in the state space. The solution to the above Bellman recursion is hard to compute [14] and closed-form exists just for few problems [10]. For a survey on strategies to approximate the Bellman recursion we refer to [14, 13].

As the worst-case cost-to-go (6.8) is hard to compute over the entire state space, we defined the worst-case cost-to-go over the safe set as

$$L^j_{\pi^j}(x) = \begin{cases} \max\limits_{w \in \mathcal{W}} [h(x, \pi^j(x)) + L^j_{\pi^j}(Ax + B\pi^j(x) + w))] & \text{If } x \in \mathcal{SS}^j \\ +\infty & \text{If } x \notin \mathcal{SS}^j \end{cases} \tag{6.9}$$

Notice that, for all $x \in \mathcal{SS}^j$, the above function coincides with the Bellman equation (6.8). The difference between $J^j_{\pi^j}(\cdot)$ and $L^j_{\pi^j}(\cdot)$ is that the domain of the latter is the safe set $\mathcal{SS}^j$ from (6.6). This fact allows us to use a simple data-based strategy to approximate $L^j_{\pi^j}(\cdot)$, as shown in Section 6.5.2.

Finally, for all $x \in \mathcal{CS}^j$ we define the function

$$Q^j(x) = \min_{\mu} \mu \mid (x, \mu) \in \text{conv}\big(\textstyle\bigcup_{k=0}^{j} \text{epi}(L_{\pi^j}(x)^j) \big), \tag{6.10}$$

which interpolates the worst-case cost-to-go functions $L^k_{\pi^k}(\cdot)$ for $k \in \{0, \dots, j\}$. Notice that the above $Q^j(\cdot)$ is simply a convexification of the cost-to-go functions (i.e. $\text{epi}(Q^j(x)) = \text{conv}\big(\cup_{k=0}^{j} \text{epi}(L_{\pi^k}(x)^k) \big)$). Furthermore, if the control policies $\pi^k(\cdot)$ for $k \in \{0, \dots, j\}$ safely steer the system to the neighborhood of the origin $\mathcal{O}$, then the approximated value function $Q^j(\cdot)$ is a robust control Lyapunov function over the convex safe set $\mathcal{CS}^j$ for system (6.1), as shown by the following proposition.

Proposition 5 *For $j \geq 1$, let $\pi^j(\cdot) : \mathcal{F}^j \to \mathcal{U}$ be a control policy defined over $\mathcal{F}^j \subseteq \mathcal{X}$. Consider system (6.1) in closed-loop with $\pi^j(\cdot)$ and assume that $\forall x^j_0 \in \mathcal{F}^j$ we have $x^j_t \in \mathcal{X}$ and $x^j_{T^j} \in \mathcal{O}$ $\forall w_t \in \mathcal{W}, k \geq 0$. Then, $Q^j(\cdot)$ is a robust control Lyapunov function, i.e.*

$$\min_{u \in \mathcal{U}} \max_{w \in \mathcal{W}} Q^j(Ax + Bu + w) + h(x, u) - Q^j(x) \leq 0$$

for all $x \in \mathcal{CS}^j$.

Proof *By the assumptions on $\pi^k(\cdot)$ for $k \in \{0,\dots,j\}$ and definitions* (6.9) *and* (6.10)*, we have that $\forall x^k \in \mathcal{SS}^k, k \in \{0,\dots,j\}$*

$$\max_{w\in\mathcal{W}} Q^j(Ax^k + Bu^k + w) + h(x,u^k)) - Q^j(x^k) \leq 0 \tag{6.11}$$

for $u^k = \pi^k(x^k)$. From (6.10) *we have that if $x \in \mathcal{CS}^j$, then we can find some multiplies $\lambda^k \geq 0$ for $k \in \{0,\dots,j\}$ such that $\sum_{k=0}^j \lambda^k = 1$, $\sum_{k=0}^j \lambda^k x^k = x$ and $\sum_{k=0}^j \lambda^k Q^j(x^k) = Q^j(x)$. Now, we notice that by the Assumption 8 and* (6.11) *we have that $\forall x \in \mathcal{CS}^j$*

$$\begin{aligned} Q^j(x) &= \textstyle\sum_{k=0}^j \lambda^k Q^j(x^k) \\ &\geq \textstyle\sum_{k=0}^j \lambda^k \max_{w\in\mathcal{W}} [Q^j(Ax^k + Bu^k + w) + h(x^k,u^k)] \\ &\geq \max_{w\in\mathcal{W}} Q^j(Ax + Bu + w) + h(x,u) \end{aligned}$$

for $u = \sum_{k=0}^j \lambda^k u^k = \sum_{k=0}^j \lambda^k \pi^k(x^k) \in \mathcal{U}$. ■

6.4 Control Design

In this section, we illustrate the controller design which exploits the convex safe set (6.7) and the approximated value function (6.10). Afterwards, we describe the controller properties. In particular, we show that the proposed strategy guarantees recursive robust constraint satisfaction and iterative worst-case performance improvement.

6.4.1 Robust Learning Model Predictive Control

At each time t of the jth iteration, we design and solve the following finite time optimal control problem

$$\begin{aligned} J^{\text{LMPC},j}_{t\to t+N}(x^j_t) = \min_{\boldsymbol{\pi}^j_t(\cdot)} \max_{\bar{\mathbf{w}}^j_t} \quad & [\sum_{k=t}^{t+N-1} h(x^j_{k|t}, u^j_{k|t}) + Q^{j-1}(x^j_{t+N|t})] \\ & x^j_{k+1|t} = Ax^j_{k|t} + Bu^j_{k|t} + \bar{w}^j_{k|t}, \\ & x^j_{k|t} \in \mathcal{X}, u^j_{k|t} \in \mathcal{U}, \\ & x^j_{t+N|t} \in \mathcal{CS}^{j-1}, \\ & u^j_{k|t} = \pi^j_{k|t}(x^j_{k|t}) \\ & x^j_{t|t} = x^j_t, \\ & \forall k \in \{t,\dots,t+N\}, \forall \bar{w}^j_{k|t} \in \mathcal{W} \end{aligned} \tag{6.12}$$

where the control policy $\boldsymbol{\pi}^j_t(\cdot) = [\pi^j_{t|t}(\cdot),\dots,\pi^j_{t+N|t}(\cdot)]$ and the disturbance $\bar{\mathbf{w}}^j_t = [\bar{w}^j_{t|t},\dots,\bar{w}^j_{t+N|t}]$. The optimal feedback policy from the above finite time optimal control problem safely steers system (6.1) from x^j_t to the convex safe set, while minimizing the worst-case cost.

Let

$$\boldsymbol{\pi}_t^{j,*}(\cdot) = [\pi_{t|t}^{j,*}(\cdot), \dots, \pi_{t+N|t}^{j,*}(\cdot)] \tag{6.13}$$

be the optimal feedback policy to Problem (6.12). Then we apply to system (6.1)

$$\pi^j(x_t^j) = \pi_{t|t}^{j,*}(x_t^j). \tag{6.14}$$

The finite time optimal control problem (6.12) is solved at time $t+1$, based on the new state $x_{t+1|t+1}^j = x_{t+1}^j$, yielding a moving or receding horizon control strategy.

Furthermore, we define the domain of the LMPC policy (6.14), which is given by

$$\mathcal{F}^j = \left\{ x_0 \in \mathcal{X} \middle| \begin{array}{c} \exists \kappa(\cdot) : \mathbb{R}^n \to \mathbb{R}^d, x_t \in \mathcal{X}, \kappa(x_t) \in \mathcal{U} \\ x_{t+1} = Ax_t + B\kappa(x_t) + w_t, \\ x_N \in \mathcal{CS}^{j-1}, \forall w_t \in \mathcal{W}, t \in \{0, \dots, N\} \end{array} \right\}. \tag{6.15}$$

The set $\mathcal{F}^j$ collects the feasible initial conditions for Problem (6.12) and it is used to compute the initial state x_0^j of the jth iteration. In particular, the initial condition at the jth iteration is computed solving the following convex optimization problem,

$$x_0^j = \underset{x \in \mathcal{F}^j}{\operatorname{argmax}} \{ax \mid a^\perp x = 0\} \tag{6.16}$$

where the user-defined row vector $a \in \mathbb{R}^n$ represents the direction in which the LMPC explores the state space. Finally, $a^\perp \in \mathbb{R}^n$ in (6.16) is a row vector perpendicular to a.

It is well-known that the solution to Problem (6.12) can be computed enumerating the vertices of the disturbance over the prediction horizon [80]. Therefore, the computational complexity of Problem (6.12) explodes with the horizon length N. For this reason, it is important to construct a terminal set and terminal cost, which allow us to guarantee safety and performance improvement independently on the prediction horizon length. In the result section, we show that the proposed controller is able to safely explore the state space and to improve its performance, even with a short prediction horizon.

6.4.2 Properties

As discussed in Propositions 4-5, for every point in $\mathcal{CS}^j$ there exists a control policy which safely steers the system to the terminal goal set. The properties of $\mathcal{CS}^j$ and $Q^j(\cdot)$ allow us to guarantee that the proposed strategy meets the requirements from Section 6.1. The following theorem shows that the LMPC (6.12) and (6.14) satisfies state and input constraints while steering the system to the neighborhood of the origin $\mathcal{O}$.

Theorem 14 *Consider system* (6.1) *in closed-loop with the LMPC* (6.12) *and* (6.14)*. Let Assumptions 7-8 hold, initialize* $\mathcal{CS}^0 = \mathcal{O}$ *and* $Q^0(\cdot) = 0$*. If* $x_0^j \in \mathcal{F}^j, \forall j \geq 1$*, then the LMPC* (6.12) *and* (6.14) *is feasible for all* $t \geq 0$ *and iteration* $j \geq 1$*. Furthermore, the closed-loop system asymptotically converges to* $\mathcal{O}$*, regardless of the disturbance realization.*

Proof *Assume that at the jth iteration $Q^j(\cdot)$ is a robust control Lyapunov function defined on the robust control invariant set $\mathcal{CS}^j$. Then, by standard MPC arguments and the assumption on $x_0^j \in \mathcal{F}^j$, we have that at iteration $j+1$ the LMPC* (6.12) *and* (6.14) *recursively satisfies state and input constraints, and the closed-loop system* (6.1) *and* (6.14) *converges asymotically to the terminal set $\mathcal{O}$ [10]. Consequently, the LMPC policy at iteration $j+1$ used to compute $Q^{j+1}(\cdot)$ and $\mathcal{CS}^{j+1}$ satisfies the assumptions in Propositions 4-5, and therefore $Q^{j+1}(\cdot)$ is a robust control Lyapunov function defined on the robust control invariant set $\mathcal{CS}^{j+1}$.*
The proof is completed by induction. We initialized $Q^0(\cdot) = 0$, which is a robust control Lyapunov function defined on the robust control invariant set $\mathcal{CS}^0 = \mathcal{O}$. Therefore it follows that $\forall j \geq 1$ the LMPC (6.12) *and* (6.14) *recursively satisfies state and input constraints, and the closed-loop system* (6.1) *and* (6.14) *converges asymotically to the terminal set $\mathcal{O}$.*

Next, we discuss the performance improvement properties. In particular, we show that if the initial condition of two subsequent iterations does not change (i.e. $x_0^j = x_0^{j+1}$), then the worst-case cost iteration cost is non-increasing.

Theorem 15 *Consider system* (6.1) *in closed-loop with the LMPC* (6.12) *and* (6.14). *Let Assumptions 7-8 hold, initialize $\mathcal{CS}^0 = \mathcal{O}$ and $Q^0(\cdot) = 0$. If the initial condition of two subsequent iterations are equal, $x_0^{j+1} = x_0^j \in \mathcal{F}^j$. Then, the worst-case iteration cost* (6.2) *is non-increasing with the iteration index $J^{j+1}_{0\to T^{j+1}}(x_0^{j+1}) \leq J^j_{0\to T^j}(x_0^j)$.*

Proof *By Theorem 14, the LMPC* (6.12) *and* (6.14) *is feasible at time t of the jth iteration. Let* (6.13) *be the optimal policy time t of the jth iteration, by Proposition 5 we have*

$$\begin{aligned}
J^{LMPC,j}_{t\to t+N}(x_t^j) &= \sum_{k=t}^{t+N-1} h(x^{j,*}_{k|t}, \pi^{j,*}_{k|t}(x^{j,*}_{k|t})) + Q^{j-1}(x^{j,*}_{t+N|t}) \\
&\geq h(x^{j,*}_{t|t}, u^{j,*}_{t|t}) + \sum_{k=t+1}^{t+N-1} h(x^{j,*}_{k|t}, \pi^{j,*}_{k|t}(x^{j,*}_{k|t}))) \\
&+ \min_{u\in\mathcal{U}} \max_{w\in\mathcal{W}} Q^{j-1}(Ax^{j,*}_{t+N|t} + Bu + w) + h(x^{j,*}_{t+N|t}, u) \\
&\geq h(x^{j,*}_{t|t}, u^{j,*}_{t|t}) + \min_{\boldsymbol{\pi}_t^j(\cdot)} \max_{\mathbf{w}_t^j} \; [\sum_{k=t}^{t+N-1} h(x_{k|t}, u_{k|t}) + Q^{j-1}(x_{t+N|t})] \\
&= h(x^{j,*}_{t|t}, u^{j,*}_{t|t}) + J^{LMPC,j}_{t+1\to t+1+N}(x^j_{t+1}).
\end{aligned}$$

The above equation and the convergence of the closed-loop system (6.1) *and* (6.14) *from Theorem 1 imply that*

$$\begin{aligned}
J^{LMPC,j}_{0\to N}(x_0^j) &\geq h(x^{j,*}_{0|0}, x^{j,*}_{0|0}) + J^{LMPC,j}_{1\to 1+N}(x^j_{t+1}) \\
&\geq \sum_{t=0}^{\infty} h(x^{j,*}_{t|t}, u^{j,*}_{t|t}) + \lim_{t\to\infty} J^{LMPC,j}_{t\to t+N}(x_t^j) = \sum_{t=0}^{\infty} h(x_t^j, u_t^j).
\end{aligned}$$

The above derivation holds for all disturbance realization, therefore we have that

$$J^{LMPC,j}_{0\to N}(x^j_0) \geq J^j_{\pi^j}(x^j_0).$$

Finally we notice that the above inequality together with Equations (6.9)-(6.10) *and the feasibility of the LMPC policy* $\pi^j(\cdot)$ (6.14) *at the next iteration* $j+1$ *imply that*

$$\begin{aligned} J^j_{\pi^j}(x^j_0) &= L^j_{\pi^j}(x^j_0) \\ &= \max_{w^j_0,\dots,w^j_{N-1}} \sum_{k=0}^{N-1} [h(x^j_t, \pi^j(x^j_t)) + L^j_{\pi^j}(x^j_N)] \\ &\geq \max_{w^j_0,\dots,w^j_{N-1}} \sum_{k=0}^{N-1} [h(x^j_t, \pi^j(x^j_t)) + Q^j(x^j_N)] \\ &\geq J^{LMPC,j+1}_{0\to N}(x^j_0) \geq J^{j+1}_{\pi^{j+1}}(x^j_0) = J^{j+1}_{\pi^{j+1}}(x^{j+1}_0). \end{aligned}$$

Finally, we show that the domain of the LMPC (6.12) and (6.14) does not shrink at each iteration.

Theorem 16 *Consider system* (6.1) *in closed-loop with the LMPC* (6.12) *and* (6.14)*. Let Assumptions 7-8 hold, and initialize* $\mathcal{CS}^0 = \mathcal{O}$ *and* $Q^0(\cdot) = 0$*. If* $x^j_0 \in \mathcal{F}^j, \forall j \geq 1$*. Then, the domain of which the LMPC defined in* (6.15) *does not shrink at each iteration, i.e.* $\mathcal{F}^i \subseteq \mathcal{F}^j, \forall j \geq i$*.*

Proof *The proof follows from the definition of the convex safe set. Notice that by definition* (6.7) *we have that* $\mathcal{CS}^i \subseteq \mathcal{CS}^j, \forall j \geq i$*. Therefore, the terminal set in* (6.15) *is not shrinking at each iteration and* $\mathcal{F}^i \subseteq \mathcal{F}^j, \forall j \geq i$*.*

6.5 Sampled Based Implementation

In this section, we show how to approximate the convex safe set $\mathcal{CS}^j$ and the value function $Q^j(\cdot)$. At each jth iteration, we collect R roll-out of the closed-loop system, which are associated with R sampled disturbance sequences. Afterwards, we exploit these stored trajectories to approximate the robust reachable sets (6.5) and the worst-case cost-to-go (6.9).

6.5.1 Sample-Based Convex Safe Set

We define the ith disturbance realization sequence $\mathbf{w}^j_i = [w^j_{0,i}, \dots, w^j_{T^j,i}]$, where $w^j_{t,i}$ is the realized disturbance at time t of the jth iteration. Furthermore, we denote the stored closed-loop trajectory associated with the ith disturbance realization $\mathbf{w}^j_i$ as

$$\mathbf{x}^j(\mathbf{w}^j_i) = [x^j_0(\mathbf{w}^j_i), \dots, x^j_{T^j}(\mathbf{w}^j_i)], \tag{6.17}$$

where T^j is the time at which the terminal goal set $\mathcal{O}$ is reached. The above notation emphasizes that the realized state $x_t^j(\mathbf{w}_i^j)$ is a function of the realized disturbance sequence $\mathbf{w}_i^j$. Now, we notice that at each time t of the jth iteration the state $x_t^j(\mathbf{w}_i^j)$ is contained into the t-steps robust reachable set from x_0^j (i.e. $x_t^j(\mathbf{w}_i^j) \in \mathcal{R}_t(x_0^j)$). Therefore, we approximate the t-steps robust reachable set $\mathcal{R}_t(x_0^j)$ using R roll-outs. In particular, for $i \in \{1, \dots, R\}$ sampled disturbance sequences $\mathbf{w}_i^j$ we define the approximated t-steps robust reachable set

$$\tilde{\mathcal{R}}_t(x_0^j) = \text{conv}\left(\bigcup_{i=1}^{R} x_t^j(\mathbf{w}_i^j)\right) \subseteq \mathcal{R}_t(x_0^j). \tag{6.18}$$

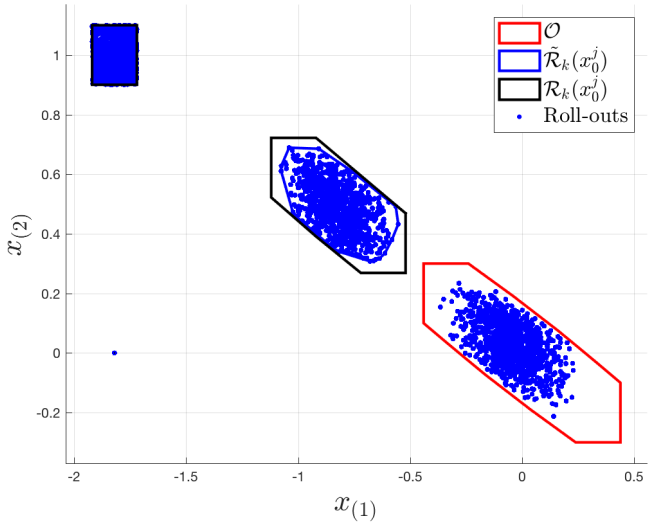


Figure 6.2: Approximated robust reachable sets $\tilde{\mathcal{R}}_t$ from (6.18) construct using 1000 roll-outs. We notice that the approximated robust reachable sets $\tilde{\mathcal{R}}_t$ are an inner approximation the robust reachable sets $\mathcal{R}_t$ from (6.5).

Finally, we define the approximated safe set

$$\tilde{\mathcal{SS}}^j = \left\{\bigcup_{t=0}^{T^j} \tilde{\mathcal{R}}_t(x_0^j)\right\} \bigcup \mathcal{O},$$

which is used to construct the approximated convex safe set,

$$\tilde{\mathcal{CS}}^j = \text{conv}\left(\bigcup_{k=0}^{j} \tilde{\mathcal{SS}}^k\right). \tag{6.19}$$

It is important to underline that the above approximated convex safe set $\tilde{\mathcal{CS}}^j$ is not invariant, as the approximated reachable sets are an inner approximation of the exact reachable sets

in Figure 6.2. Indeed, it may exist a disturbance realization which can steer the closed-loop system (6.1) and (6.14) outside $\tilde{\mathcal{CS}}^j$. In particular, given $x \in \tilde{\mathcal{CS}}^j$ there is a probability $\epsilon > 0$ that the closed-loop system evolves outside $\tilde{\mathcal{CS}}^j$,

$$\Pr(Ax + B\pi^j(x) + w \notin \tilde{\mathcal{CS}}^j | x \in \tilde{\mathcal{CS}}^j) \geq \epsilon. \tag{6.20}$$

In the result section, we show that the above probability is a function of the number of roll-outs used to construct $\tilde{\mathcal{CS}}^j$. In particular as more roll-outs are collected, $\tilde{\mathcal{CS}}^j$ from (6.19) better approximates the convex safe set $\mathcal{CS}^j$ from (6.7).

6.5.2 Sample-Based Q-function

Similarly, we exploit the closed-loop data to approximate the cost-to-go function $L^j_{\pi^j}(\cdot)$ in (6.9). First, we define the realized cost-to-go associated with the stored state $x^j_t(\mathbf{w}^i) \in \tilde{\mathcal{R}}_t(x^j_0) \subseteq \tilde{\mathcal{SS}}^j$,

$$\tilde{J}^j_{k\to T^j}(x^j_t(\mathbf{w}^i)) = \sum_{t=k}^{T^j} h\Big(x^j_t(\mathbf{w}^i), \pi^j(x^j_t(\mathbf{w}^i))\Big). \tag{6.21}$$

The realized cost (6.21) associated with the realized trajectory (6.17) is used to approximate the worst-case cost-to-go function $L^j_{\pi^j}(\cdot)$, over the approximated robust reachable set $\tilde{\mathcal{R}}_t(x^j_0)$ from (6.18). First, we compute an hyperplane which upper-bounds the realized cost $\tilde{J}^j_{k\to T^j}(x^j_t(\mathbf{w}^i))$ for all stored states $\big\{\bigcup_{i=1}^M x^j_t(\mathbf{w}^i)\big\} \in \tilde{\mathcal{R}}_t(x^j_0)$. In particular, for time t of the jth iteration we defined the hyperplane $a^j_t x + b^j_t$, where

$$\begin{aligned} [a^j_t, b^j_t] = \underset{a\in\mathbb{R}^n, b\in\mathbb{R}}{\operatorname{argmin}} \quad & \sum_{i=0}^{M} ||a x^j_t(\mathbf{w}^i) + b - \tilde{J}^j_{t\to T^j}(x^j_t(\mathbf{w}^i))||_2^2 \\ \text{s.t.} \quad & a x^j_t(\mathbf{w}^i) + b \geq \tilde{J}^j_{t\to T^j}(x^j_t(\mathbf{w}^i)),\ \forall i \in \{0,\dots,M\}. \end{aligned} \tag{6.22}$$

Finally at the jth iteration, the hyperplanes $a^j_t x + b^j_t$ are used to approximate the worst-case cost-to-go $L^j_{\pi^j}(\cdot)$ from (6.9) as follows,

$$\tilde{L}^j_{\pi^j}(x) = \begin{cases} +\infty & \text{If } x \notin \tilde{\mathcal{SS}}^j \\ 0 & \text{Elseif } x \in \mathcal{O} \\ \min\limits_{k\in\mathcal{K}^j} \mathbb{1}_{\tilde{\mathcal{R}}_t(x)}(a^j_t x + b^j_t) & \text{Else} \end{cases}, \tag{6.23}$$

where the set $\mathcal{K}^j = \{0,\dots,T^j\}$ and the indicator function $\mathbb{1}_{\mathcal{S}}(x) = x$ is defined over the set $\mathcal{S}$. The resulting approximated value function is defined as

$$\tilde{Q}^j(x) = \min_{\mu} \mu \mid (x,\mu) \in \bigcup_{k=0}^{j} \operatorname{conv}\big(\operatorname{epi}(\tilde{L}_{\pi^j}(x)^j)\big). \tag{6.24}$$

Finally, we underline that the above approximated value function is not a control Lyapunov function for system (6.1). Indeed, there is a probability $\gamma > 0$ that Equation (6.11) does not hold and $\tilde{Q}^j(\cdot)$ is not decreasing along the closed-loop trajectory,

$$\Pr\big(\tilde{Q}^j(Ax + B\pi^j(x) + w) + h(x, \pi^j(x)) - \tilde{Q}^j(x) > 0\big) \geq \gamma. \tag{6.25}$$

In the result section, we show that above probability is inversely proportional to the number R of realized trajectories used to construct $\tilde{L}^j_{\pi^j}(\cdot)$ from (6.23).

6.6 Examples

Finally, we test the proposed strategy on a double integrator and on the parking problem from [13, Section 4]. We show that in both examples the LMPC is able to improve the closed-loop performance while guarantees state and input constraints.

6.6.1 Constrained LQR Problem

We test the proposed control strategy on the following double integrator system

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t + w_t, \tag{6.26}$$

where the the random disturbance w_t is uniformly distributed on the set $\mathcal{W} = \{w \in \mathbb{R}^2 : ||w_t||_\infty \leq 0.1\}$. The system is subjected to the following state and input constraints, $x_t \in \mathcal{X} = \{x \in \mathbb{R}^2 : ||x||_\infty \leq 10\}$ and $u_t \in \mathcal{U} = \{u \in \mathbb{R}^2 : ||u||_\infty \leq 1\}$, for all $t \geq 0$. Furthermore, we compute the minimal robust positive invariant set $\mathcal{O}$ for the autonomous system $x_{t+1} = (A + BK)x_t + w_t$ where $-K$ is the LQR gain for $Q = 1$ and $R = 1$. Finally, we define the stage cost $h(x, u) = |x|_\mathcal{O} + |u|_{KO}$ which satisfies Assumption 8.

The convex safe set $\mathcal{CS}^j$ and value function $Q^j(\cdot)$, used in the LMPC (6.12) and (6.14), are approximated as described in Section 6.5. In particular at each iteration j, we use R roll-outs to compute the approximated safe set $\tilde{\mathcal{CS}}^j$ and value function $\tilde{Q}^j(\cdot)$. In order to initialize the LMPC we set $N = 3$, $\tilde{\mathcal{CS}}^0 = \mathcal{O}$ and $\tilde{Q}^0(\cdot) = 0$. Finally at each jth iteration, the initial state x^j_0 is computed as the furthest point along the negative x-axis which belongs to $\mathcal{F}^j$. Basically, we set $a = [-1,\ 0]$ in (6.16).

6.6.2 Convex Safe Set and Value Function Approximation

In this section, we construct the convex safe set $\tilde{\mathcal{CS}}^1$ and value function approximation $\tilde{Q}^1(\cdot)$ using $R = 100$ and $R = 1000$ roll-outs. Furthermore, we perform 1000 Monte-Carlo closed-loop simulations to estimate the properties of $\tilde{\mathcal{CS}}^1$ and $\tilde{Q}^1(\cdot)$.

Figure 6.3 shows the terminal set $\mathcal{O}$ and the approximated robust reachable sets $\tilde{\mathcal{R}}_t(x^1_0)$, which are used to construct the approximated convex safe set $\tilde{\mathcal{CS}}^j$ with $R = 100$ and

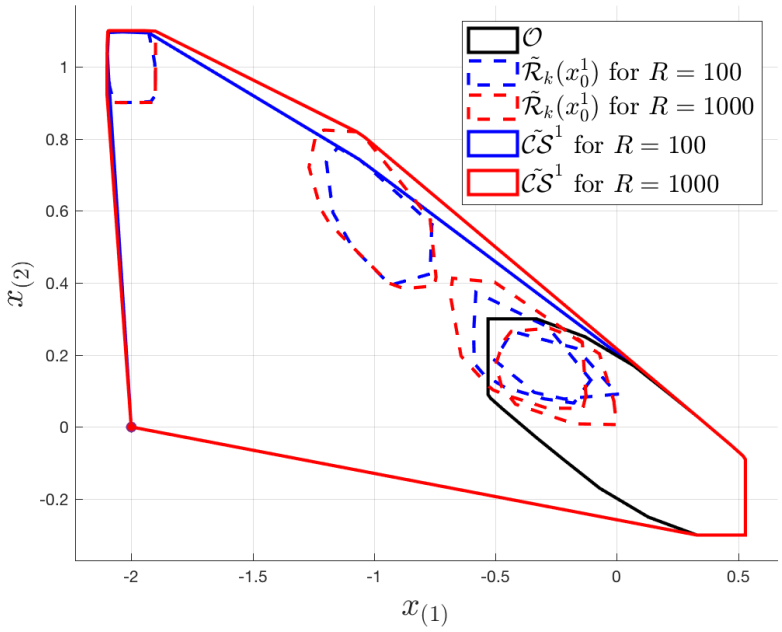


Figure 6.3: The approximated robust reachable sets $\tilde{\mathcal{R}}_t$ (6.18) used to construct $\tilde{\mathcal{CS}}^1$ with $R = 100$ and $R = 1000$ roll-outs. Notice that the approximated convex safe set $\tilde{\mathcal{CS}}^1$ constructed using 1000 roll-outs contains the one constructed using 100.

$R = 1000$ roll-outs. As expected, the approximated convex safe set $\tilde{\mathcal{CS}}^j$ constructed using 1000 trajectories contains the one constructed using 100 trajectories. As mentioned in Section 6.5.1 (Eq. (6.20)), the approximated convex safe set is not invariant. Indeed, there is a probability $\epsilon > 0$ that, given a state $x \in \tilde{\mathcal{CS}}^1$, the closed-loop system evolves outside $\tilde{\mathcal{CS}}^1$. In order to estimate the probability ϵ, we perform 1000 Monte-Carlo simulations for the closed-loop system (6.1) and (6.14) and we compute the percentage of realized states which evolved outside $\tilde{\mathcal{CS}}^j$. As expected the probability ϵ decreases as more roll-outs are used to construct $\tilde{\mathcal{CS}}^1$. In particular, we have that $\epsilon \sim 3.6\%$ and $\epsilon \sim 0.3\%$ for $R = 100$ and $R = 1000$, respectively.

Finally, we analyze how the number of roll-outs affects the approximated value function $\tilde{Q}^1(\cdot)$. Figure 6.4 shows the approximated value function $\tilde{Q}^1(\cdot)$ constructed with $R = 100$ and $R = 1000$ roll-outs. First, we notice that the domain of approximated value function $\tilde{Q}^1(\cdot)$ is enlarged as more realized trajectories are used to compute the approximation. Indeed, the domain of $\tilde{Q}^1(\cdot)$ is the approximated safe set $\tilde{\mathcal{CS}}^1$ from Figure 6.3. Second, we recall that $\tilde{Q}^1(\cdot)$ is constructed based on sampled disturbance sequences and it underestimates $Q^1(\cdot)$, which considers the whole disturbance support. Therefore, we expect that as more sample disturbance sequences are considered $\tilde{Q}^1(\cdot)$ better approximates $Q^1(\cdot)$. This intuition is confirmed by Figure 6.4, we notice that $\tilde{Q}^1(\cdot)$ constructed with 1000 trajectories

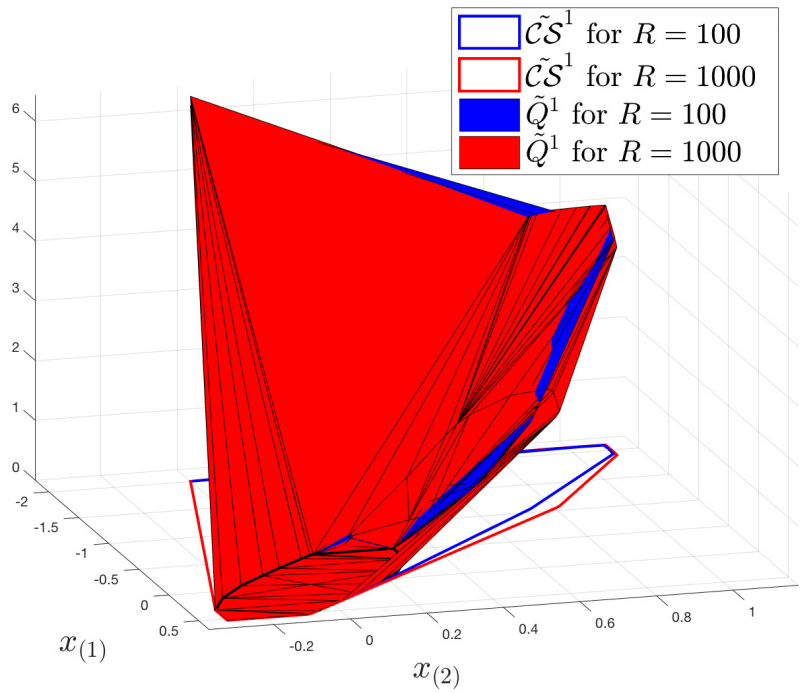


Figure 6.4: Approximated value function $\tilde{Q}^j(\cdot)$ constructed with $R = 100$ and $R = 1000$ roll-outs. Note that as more trajectories are used the value of $\tilde{Q}^j(\cdot)$ increases almost everywhere, thus it better approximated $Q^j(\cdot)$.

upper-bounds almost everywhere the value function $\tilde{Q}^1(\cdot)$ constructed with 100 trajectories, therefore it better approximates $Q^1(\cdot)$. Finally, we recall from Equation (6.25) that $\tilde{Q}^1(\cdot)$ is not a robust control Lyapunov function. Indeed, there is a probability $\gamma > 0$ that $\tilde{Q}^1(\cdot)$ is not decreasing along the realized closed-loop trajectory. In order to estimate the probability γ, we use 1000 Monte Carlo simulations. As expected, the probability γ decreases as more closed-loop trajectories are used to construct $\tilde{Q}^1(\cdot)$. In particular, we have $\gamma \sim 10.1\%$ and $\gamma \sim 4.3\%$ for $R = 100$ and $R = 1000$, respectively.

6.6.3 Iterative Policy Update

In this section we run the LMPC for 10 iterations. In particular, at each jth iteration we collect $R = 1000$ roll-outs which are used to compute the approximated convex safe set $\tilde{\mathcal{CS}}^j$ and the approximated value function $\tilde{Q}^j(\cdot)$. We show that the LMPC is able to explore the state space while safely steering the system to the terminal set $\mathcal{O}$.

As stated in Section 6.6, at each jth iteration we compute the initial condition x_0^j as the furthest point along the negative x-axis such that Problem (6.12) is feasible. Notice that by Theorem 16, the domain of the LMPC policy $\mathcal{F}^j$ is enlarged at each iteration (i.e. $\mathcal{F}^k \subseteq \mathcal{F}^j$ for all $k \in \{1, \dots, j\}$). As a result, the region of the state space from which the

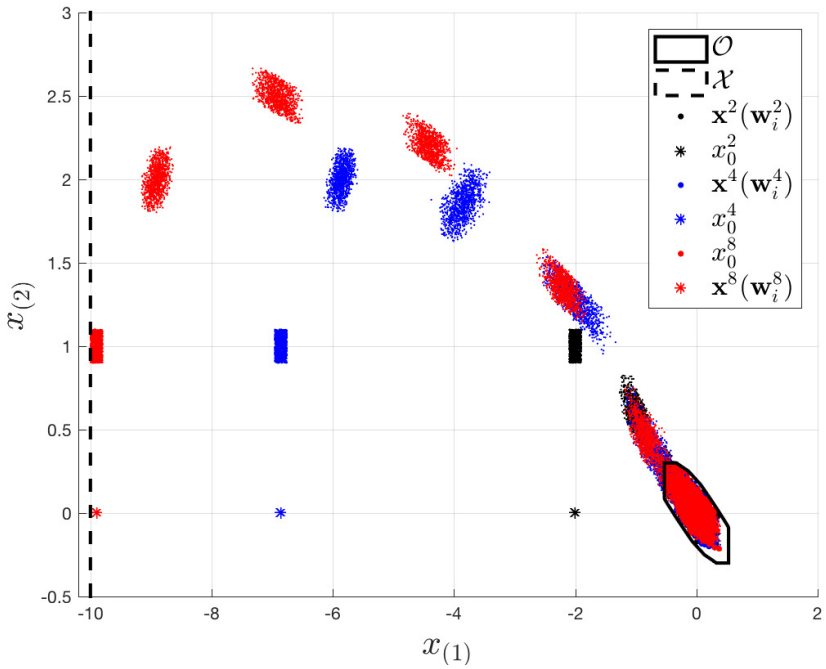


Figure 6.5: For iterations $j \in \{2, 4, 8\}$ and $i = \{1, \ldots, 1000\}$ disturbance realizations we show the closed-loop trajectories $\mathbf{x}^j(\mathbf{w}_i^j)$ from (6.17). Furthermore, we report the initial condition x_0^j which is further from the origin at each iteration.

controller is able to safely complete the control task grows at each iteration. This fact is highlighted in Table 6.1, where we report the initial condition x_0^j as a function of the iteration index. Furthermore, in Figure 6.5 we show 1000 realized trajectories for the 2nd, 4th and 8th iterations. We notice that at each iteration the LMPC safely operates the system over progressively larger regions of the state space, until the closed-loop trajectory is close to saturate the state constraints.

Table 6.1: Initial condition x_0^j at each jth iteration.

$x_0^1 = -\begin{bmatrix}2.00 & 0\end{bmatrix}^\top$	$x_0^6 = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$
$x_0^2 = -\begin{bmatrix}5.46 & 0\end{bmatrix}^\top$	$x_0^7 = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$
$x_0^3 = -\begin{bmatrix}6.86 & 0\end{bmatrix}^\top$	$x_0^8 = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$
$x_0^4 = -\begin{bmatrix}9.35 & 0\end{bmatrix}^\top$	$x_0^9 = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$
$x_0^5 = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$	$x_0^{10} = -\begin{bmatrix}9.90 & 0\end{bmatrix}^\top$

Finally, in Figure 6.6 we report the approximated value function $\tilde{Q}^j(\cdot)$ for the 2nd, 4th and 8th iterations. We recall that the domain of $\tilde{Q}^j(\cdot)$ is the approximated convex safe set $\tilde{\mathcal{CS}}^j$, which is enlarged at each iteration. Therefore, as more iterations of the control task

are executed, $\tilde{Q}^j(\cdot)$ approximates the value function over larger regions of the state space, as shown in Figure 6.6.

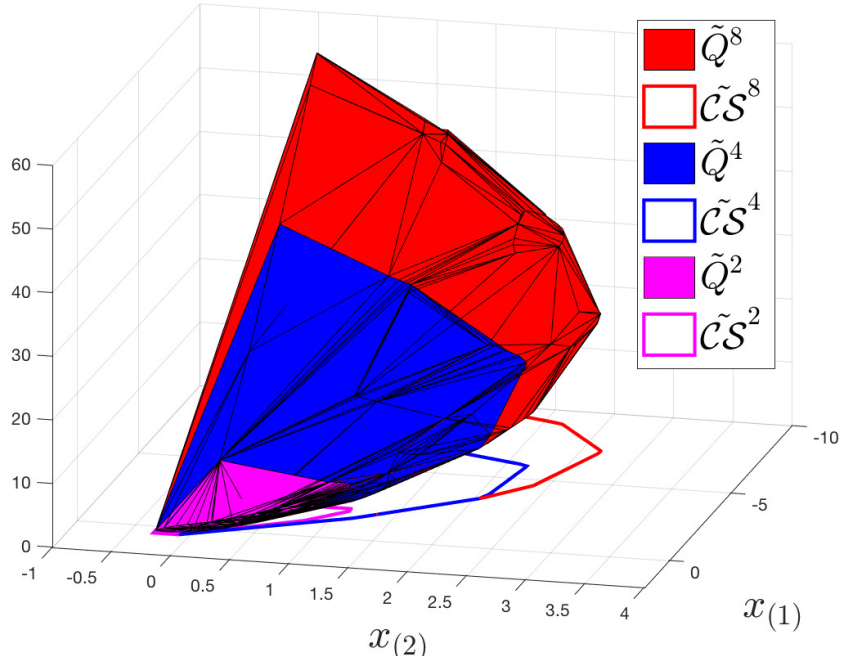


Figure 6.6: Approximated value function $\tilde{Q}^j$ at the 2nd, 4th and 8th iteration. Notice that the domain of $\tilde{Q}^j$ is enlarged at each iteration.

6.6.4 Performance Improvement

In this section we empirically validate Theorem 15. We design a LMPC which minimizes the stage cost $\bar{h}(x,u) = 0.1|x|_{\mathcal{O}} + |u|_{KO}$. Afterwards, we run the closed-loop system for 10 iterations starting from the same initial condition, $x_0^j = -[0,\ 9.9]\ \forall j \in \{0,\dots,9\}$. In order to initialize the LMPC, we use a suboptimal controller which robustly steers system (6.26) to $\mathcal{O}$ and we exploit the closed-loop data to initialize the approximated convex safe set and value function.

Figure 6.7 shows the closed-loop cost $\tilde{J}^j_{0\to T^j}(x_0^j(\mathbf{w}_i^j))$ from (6.21) and the worst-case realized cost

$$\max_{i\in\{0,\dots,R\}} \tilde{J}^j_{1\to T^j}(x_0^j(\mathbf{w}_i^j)) \tag{6.27}$$

for 10 iterations. We notice that the LMPC is able to improve the worst-case realized cost associated with the suboptimal policy used at the 0th iteration. Furthermore, we underline that the controller performs exactly the same task at each iteration ($x_0^j = x_0^i, \forall j, i \geq 0$) and the worst-case realized cost (6.27) decreases at each iteration, until it converges within a tolerance of 0.7% as stated in Theorem 15.

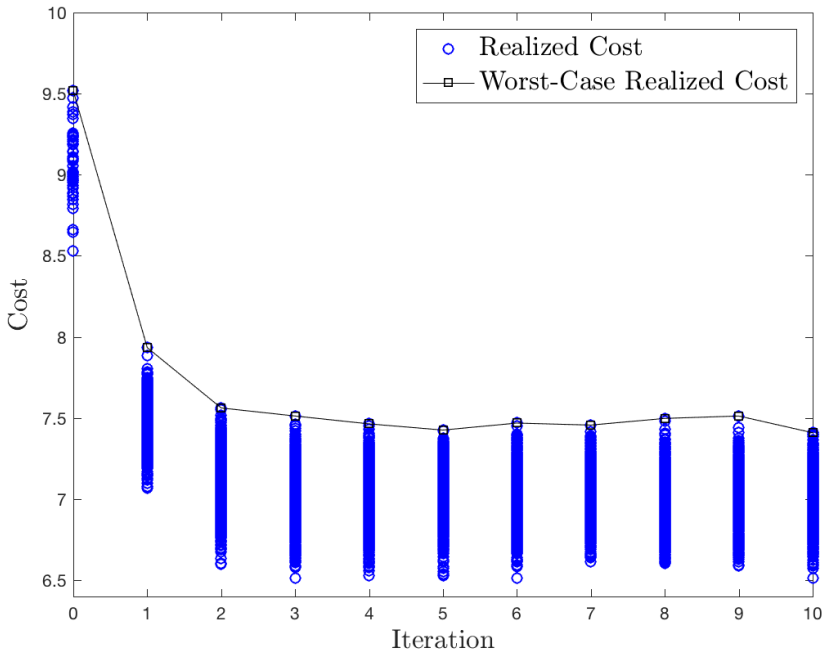


Figure 6.7: Worst-case realized cost and realized cost of the LMPC over the iteration index. We notice that the LMPC improves the worst-case realized cost from the suboptimal controller at the 0th iteration, until it reaches convergence.

6.6.5 Parking Problem

Finally, we tested the proposed strategy on the parking example from [13, Section 3]. The goal of this problem is to minimize the expected parking cost. There are $s = n + 1$ parking spaces. The the driver starts from the parking spot $s = n + 1$, and at each parking space s it may proceed to the next parking space $s - 1$ or park, if the parking spot is free. If the drivers reaches the parking spot $s = 0$ it has to park and pay a fee of \$100. Otherwise if the parking spot s is available, the driver can park by paying a fee of $\$s$. The parking spots are free with probability p and the driver knows the availability when it reaches a particular parking location.

At iteration 0 we implemented a control policy that drives the vehicle to the garage and pays \$100 to park. We performed 1000 roll-out and we used this closed-loop data to implement the LMPC. As in this example the goal is to minimize the expected cost, in (6.22) we computed the hyperplane which approximates the empirical mean over each approximated robust reachable set.

Figure 6.8 shows the steady-state LMPC policy and the associated value function. We notice that the LMPC policy overlaps with the optimal one from [13, Section 3]. Finally, Figure 6.9 reports the data points used to defined the approximated Q-function from (6.24). We notice that he stored data points used to construct the Q-function are significantly lower then the total amount of stored data points.

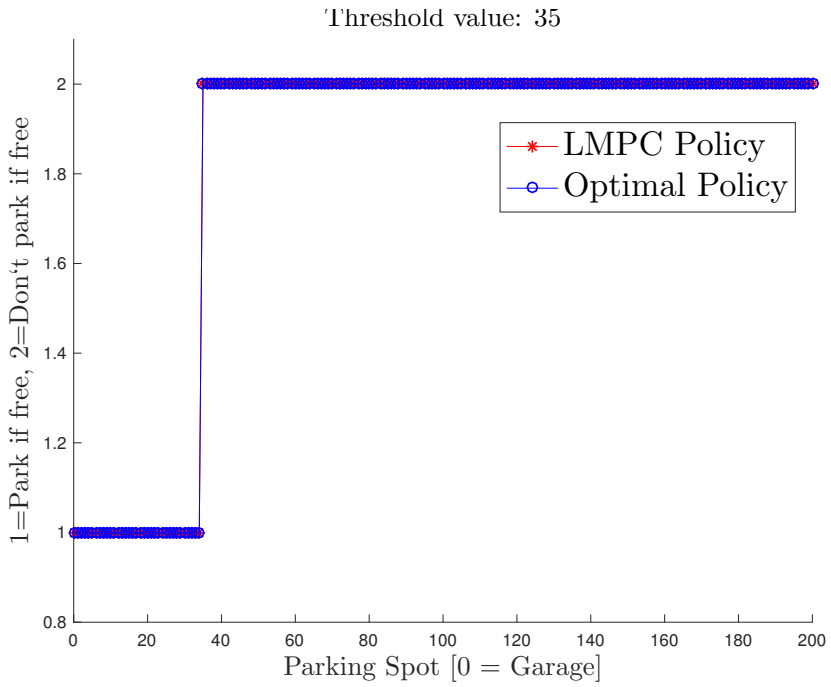


Figure 6.8: Comparison between the LMPC policy at convergence and the optimal policy from [13, Section 3].

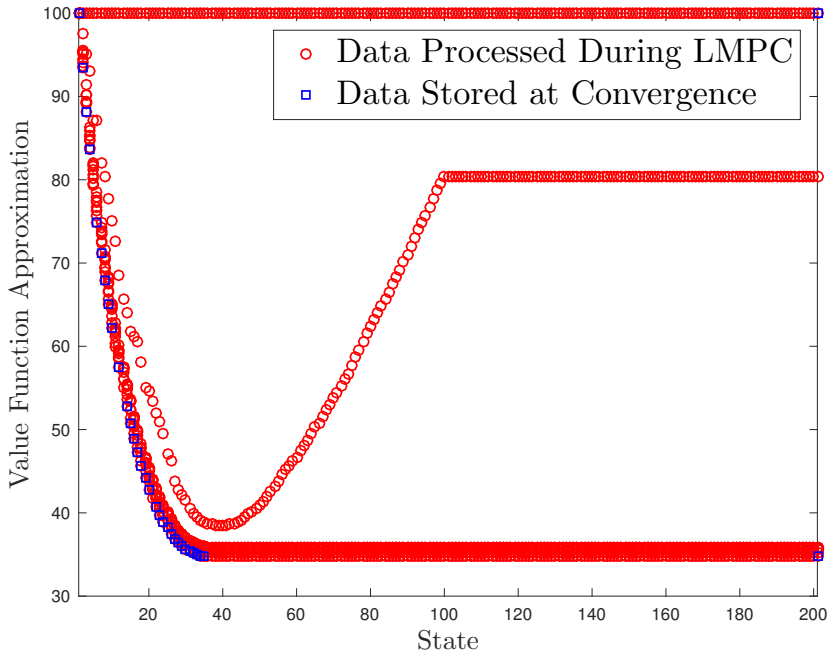


Figure 6.9: Stored data point needed to construct the approximated Q-function from (6.24) and total data points processed during the iterative process.

Chapter 7

Feedback Policy Parametrization for Robust LMPC

As we have discussed in the previous chapters, the LMPC computes the control action by solving a finite time optimal control problem over a moving time window. When uncertainty is acting on the system, the control problem is carried over a space of feedback policies. Such space should contain the safe policies, which may be used to complete the control task from any state into the safe set. If this condition is not verified, then the control design is challenging. In this chapter, we proposed an adaptive prediction horizon strategy which allows us to pick the space of feedback policies used by the LMPC independently form the safe policies. First, we illustrate how to construct robust robust sets from historical data and we characterized the associated safe control policies. Then, we propose an iterative LMPC design procedure, where data generated by a robust controller at iteration j are used to design a robust LMPC at the next $j+1$ iteration. We show that this procedure allows us to iteratively enlarge the domain of the control policy and it guarantees recursive constraints satisfaction, input to state stability and performance bounds for the certainty equivalent closed-loop system. The effectiveness of the proposed control scheme is illustrated on a linear system subject to bounded additive disturbance.

7.1 Problem Formulation

Consider the uncertain linear time-invariant system,

$$x_{t+1}^j = Ax_t^j + Bu_t^j + w_t^j \tag{7.1}$$

where $x_t^j \in \mathbb{R}^n$ and $u_t^j \in \mathbb{R}^d$ are the state and the input at time t of the jth iteration, and the matrices A and B are known. The disturbances w_t^j are zero mean independent and identically distributed (*i.i.d.*) with bounded support $\mathcal{W}$.

Assumption 9 *The disturbance's support $\mathcal{W}$ is a compact polytope described by l vertices $\{v_w^1, \dots, v_w^l\}$ and it contains the origin.*

Furthermore, system (7.1) is subject to the following convex constraints on states and inputs

$$x_t^j \in \mathcal{X} \text{ and } u_t^j \in \mathcal{U},\ \forall t \geq 0, \forall j \geq 0 \tag{7.2}$$

for $\mathcal{X}$ and $\mathcal{U}$ compact.

7.1.1 Control Design Objectives

In this section, we describe the goals of the iterative synthesis process. At each iteration j, our objective is to design a state-feedback policy for the uncertain system (7.1)

$$\pi^j(\cdot) : \mathcal{C}^j \subseteq \mathbb{R}^n \to \mathbb{R}^d, \tag{7.3}$$

such that at each jth iteration and for all $x_0^j \in \mathcal{C}^j \subseteq \mathbb{R}^n$ we have that:

1. The certainty equivalent system

$$\bar{x}_{t+1}^j = A\bar{x}_t^j + B\bar{u}_t^j \tag{7.4}$$

   with $u_t^j = \pi(x_t^j)$ converges asymptotically to goal set $\mathcal{O}$, i.e. $\lim_{t\to\infty} \bar{x}_t^j \in \mathcal{O}$.

2. The closed-loop system $x_{t+1}^j = Ax_t^j + B\pi^j(x_t^j) + w_t^j$ is Input to State Stable (ISS) with respect to the set $\mathcal{O}$ (see Section 2.2 for the definition of ISS).

3. The closed-loop state and input constraints are robustly satisfied, namely

$$x_t^j \in \mathcal{X} \text{ and } \pi^j(x_t^j) \in \mathcal{U}, \forall w_t^j \in \mathcal{W},\ \forall t \geq 0.$$

4. The domain $\mathcal{C}^j$ of policy $\pi^j(\cdot)$ does not shrink with the iteration index, i.e., $\mathcal{C}^j \subseteq \mathcal{C}^{j+1}$.

5. The iteration cost of the certainty equivalent system (7.4), defined as

$$J_{0\to\infty}^j(\bar{x}_0^j) = \sum_{k=0}^{\infty} h(\bar{x}_t^j, \pi^j(\bar{x}_t^j)),$$

   is upper-bounded by a function $Q^{j-1}(\cdot)$ (i.e. $J_{0\to\infty}^j(x_0^j) \leq Q^{j-1}(x_0^j)$), which is non-increasing at each iteration

$$Q^j(\bar{x}) \leq Q^k(\bar{x}), \forall j \geq k.$$

Property 5) implies that, as more data is collected, the upper-bound on the performance of the certainty equivalent closed-loop system is non-increasing.

Throughout this chapter we make the following assumptions.

Assumption 10 *The set $\mathcal{O} \subset \mathbb{R}^n$ is a robust positive invariant set for the autonomous system $x_{t+1} = (A - BK)x_t + w_t$ with $w_t \in \mathcal{W}$. Furthermore, $\mathcal{O}$ is a polyhedron defined through its vertices $\{v_o^1, \dots, v_o^m\}$ and*

$$K\mathcal{O} = \{u \in \mathbb{R}^d : \exists x \in \mathcal{O}, u = Kx\}.$$

.

Assumption 11 *We assume that the stage cost $h(\cdot, \cdot)$ is continuous and jointly convex in its arguments. Furthermore, we assume that $\forall x \in \mathbb{R}^n, \forall u \in \mathbb{R}^d$*

$$\alpha_x^l(|x|_{\mathcal{O}}) \leq h(x, 0) \leq \alpha_x^u(|x|_{\mathcal{O}})$$
$$\text{and } \alpha_u^l(|u|_{K\mathcal{O}}) \leq h(0, u) \leq \alpha_x^u(|u|_{K\mathcal{O}})$$

where $\alpha_x^u, \alpha_x^l, \alpha_u^u$ and $\alpha_u^l \in \mathcal{K}_\infty$.

Remark 8 *In Assumption 10 a robust invariant $\mathcal{O}$ is required. In the proposed approach $\mathcal{O}$ can be a very small neighborhood of the origin. In fact, the iterative nature of the control design will enlarge the closed-loop domain of attraction at each iteration.*

7.2 Preliminaries

The section describes how historical data can be used to build a *robust safe set* of states from which the control task can be executed. Furthermore, we define the *robust Q-function* which will be used to bound from above the performance of the proposed control strategy. Finally, for each robust safe set we construct a *safe policy* which may be used to complete the control task.

7.2.1 Robust Safe Set

We show how to iteratively construct robust control invariant sets. In particular, we run the closed-loop system at iteration j and we exploit the closed-loop trajectory to construct a robust safe set at the next iteration $j + 1$. Compared to the previous chapter, we show how to construct robust safe sets given a feedback policy with horizon N smaller than the task duration T^j.

At iteration $j = 1$, let

$$\boldsymbol{\pi}_t^1(\cdot) = [\pi_{t|t}^1(\cdot), \dots, \pi_{t+N\text{-}1|t}^1(\cdot)] \tag{7.5}$$

be a robust N-steps policy which is applied in a receding horizon fashion to system (7.1) and consider the resulting closed-loop system

$$x_{t+1}^1 = Ax_t^1 + B\pi_{t|t}^1(x_t) + w_t. \tag{7.6}$$

Furthermore, define the *robust convex safe set* $\mathcal{CS}^0 = \mathcal{O}$ at iteration $j = 0$ and assume that the following assumptions hold.

Assumption 12 *The closed-loop system* (7.6)*, starting from the initial condition* x_0^1*, reaches the robust convex safe set* $\mathcal{CS}^0 = \mathcal{O}$ *in* T^1 *steps, while robustly satisfying state and input constraints* (7.2)*.*

Assumption 13 *For all* $t \in \{0, \dots, T^1\}$*, the* N*-steps policy* $\boldsymbol{\pi}_t^1(\cdot)$ *in* (7.5) *steers the predicted closed-loop system*

$$x_{k+1|t}^1 = Ax_{k|t}^1 + B\pi_{k|t}^1(x_{k|t}^1) + w_{k|t}^1, \ \forall k = t, \dots, t+N-1$$

from the state x_t^1 *to the robust convex safe set* $\mathcal{CS}^0 = \mathcal{O}$ *in* N*-steps, while robustly satisfying state and input constraints* (7.2)*.*

Let the vectors

$$[x_0^1, \dots, x_{T^1}^1] \text{ and } [u_0^1, \dots, u_{T^1}^1] \tag{7.7}$$

collect states and inputs associated with a simulation of the closed-loop system. We notice that, by linearity of the system, any state in the convex-hull of the closed-loop trajectory in (7.7) can be robustly steered to $\mathcal{O}$. However, the convex hull of the states in (7.7) and $\mathcal{O}$ is not invariant, as it does not necessarily contain the k-steps robust reachable set $\mathcal{R}_{t\to t+k}(x_0^1)$ from the staring state x_0^1, as shown in Figure 7.1 (see Chapter 2 for the definition of robust reachable sets).

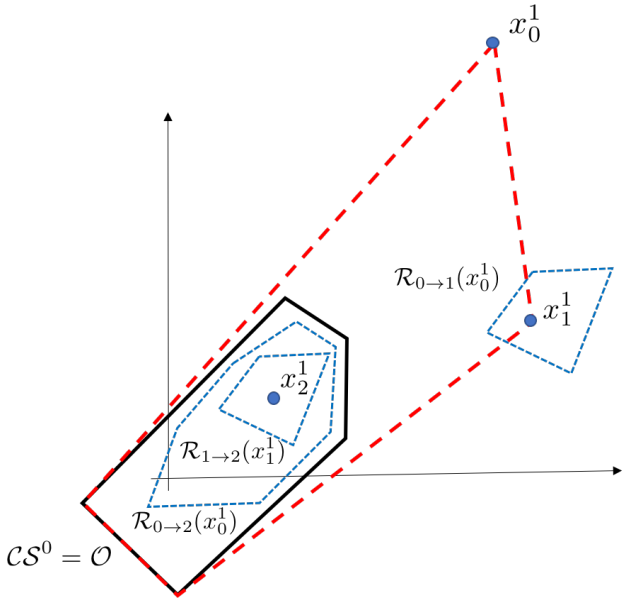


Figure 7.1: Convex-hull of the stored states and $\mathcal{O}$ (dashed red line) and the robust reachable sets $\mathcal{R}_{t\to t+k}(x_0^j)$ (dashed blue line). We notice that the convex-hull of the stored states and $\mathcal{O}$ does not contain the robust reachable sets $\mathcal{R}_{t\to t+k}(x_0^j)$ and therefore it is not a robust invariant for the closed-loop system (7.6).

Now, we notice that robust control invariant sets can be computed using k-steps robust reachable sets $\mathcal{R}_{t\to t+k}(x_0^1)$ from the stored states in (7.7). In particular, we notice that the union of the k-steps robust reachable sets $\mathcal{R}_{t\to t+k}(x_0^1)$ for $k = 0, \dots, N$ and the robust convex safe set $\mathcal{CS}^0 = \mathcal{O}$ is a robust control invariant. Therefore, we define the *robust convex safe set* at iteration $j = 1$ as the convex hull of the $N \times T^j$ robust reachable sets and the robust convex safe set $\mathcal{CS}^0$ at iteration 0,

$$\mathcal{CS}^1 = \text{Conv}\left(\left\{\bigcup_{t=0}^{T^1}\bigcup_{k=0}^{N}\mathcal{R}_{t\to t+k}(x_t^1)\right\}\bigcup\mathcal{CS}^0\right). \tag{7.8}$$

The above robust convex safe set at iteration $j = 1$ is shown in Figure 7.2.

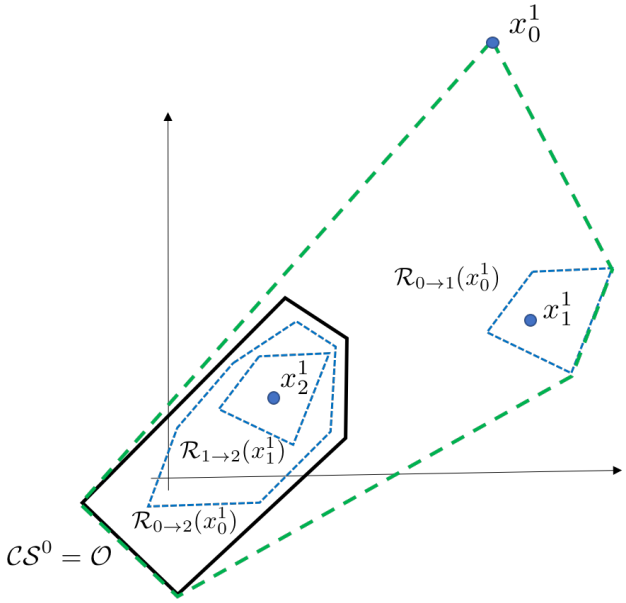


Figure 7.2: Representation of the convex safe set $\mathcal{CS}^1$ (dashed green line) and the robust reachable sets $\mathcal{R}_{t\to t+k}(x_0^1)$ (dashed blue line).

The above process is repeated at iteration j starting from data collected at iteration $j-1$. Clearly, Assumptions 12-13 must hold when $\mathcal{CS}^0$ is replaced with $\mathcal{CS}^{j-1}$ and iteration 1 with j. More formally, given the N-steps policy

$$\boldsymbol{\pi}_t^j(\cdot) = [\pi_{t|t}^j(\cdot), \dots, \pi_{t+N\text{-}1|t}^j(\cdot)] \tag{7.9}$$

and the closed-loop system

$$x_{t+1}^j = Ax_t^j + B\pi_{t|t}^j(x_t^j) + w_t^j \tag{7.10}$$

we assume that the following holds.

Assumption 14 *The closed-loop system* (7.10)*, starting from the initial condition* x_0^j*, reaches the robust convex safe set* $\mathcal{CS}^{j-1}$ *in* T^j *steps, while robustly satisfying state and input constraints* (7.2)*. Furthermore, for all* $t \in \{0, \dots, T^j\}$*, the* N*-steps policy* $\boldsymbol{\pi}_t^j$ *from* (7.9) *steers the predicted closed-loop system*

$$x_{k+1|t}^j = Ax_{k|t}^j + B\pi_{k|t}^j(x_{k|t}^j) + w_{k|t}^j, \ \forall k = t, \dots, t+N-1$$

from the state x_t^j *to the robust convex safe set* $\mathcal{CS}^j$ *in* N*-steps, while robustly satisfying state and input constraints* (7.2)*.*

Later in Section 7.3 we will show how to synthesize a control polity $\boldsymbol{\pi}_t^j$ which satisfies Assumption 14.

At iteration j, we exploits $\mathcal{CS}^{j-1}$ to iteratively define the convex safe set:

$$\mathcal{CS}^j = \text{Conv}\left(\left\{\bigcup_{t=0}^{T^j}\bigcup_{k=0}^{N} \mathcal{R}_{t\to t+k}(x_t^j)\right\}\bigcup \mathcal{CS}^{j-1}\right). \tag{7.11}$$

Details on the computation and storage of the convex safe set are provided next.

7.2.2 Robust Convex Safe Set: Vertex Representation

Recall from Assumption (10) that l denoted the number of vertices of the disturbance support. Now, we define the l^k vertices of the k-step robust reachable set $\mathcal{R}_{t\to t+k}(x_t^j, \mathcal{W})$ from x_t^j,

$$[v_{t+k|t}^{j,l}, \dots, v_{t+k|t}^{j,l^k}]. \tag{7.12}$$

The vertices of the robust reachable sets $\mathcal{R}_{t\to t+k}(x_t^i)$ for all the $k \in \{0, \dots, N\text{-}1\}$, $i \in \{0, \dots, j\}$ and $t \in \{0, \dots, T^j\}$ are collected by the following matrix

$$\begin{aligned}\mathbf{X}^j = [\mathbf{X}^{j-1},\ & v_{0|0}^{j,1}, \dots, v_{N-1|0}^{j,l^{N-1}}, \dots \\ & v_{0|t}^{j,1}, \dots, v_{N-1|t}^{j,l^{N-1}}, \dots \\ & v_{0|T^j}^{j,1}, \dots, v_{N-1|T^j}^{j,l^{N-1}}],\end{aligned} \tag{7.13}$$

where at jth iteration $v_{t+k|t}^{j,i}$ represents the ith vertex of the robust reachable sets $\mathcal{R}_{t\to t+k}(x_t^j)$. In the above recursive definition, we set $\mathbf{X}^0 = [v_o^1, \dots, v_o^m]$, where v_o^i for $i \in \{1, \dots, m\}$ are the vertices of $\mathcal{O}$ from Assumption 10.

Finally, as the columns of the matrix $\mathbf{X}^j$ in (7.13) collect all vertices of the robust reachable sets $\mathcal{R}_{t\to t+k}(x_t^j)$, the robust convex safe set $\mathcal{CS}^j$ from (7.11) can be written as

$$\mathcal{CS}^j = \left\{x \in \mathbb{R}^n : \exists \boldsymbol{\lambda}^j \geq 0, \mathbf{X}^j\boldsymbol{\lambda}^j = x \text{ and } \mathbb{1}^\top\boldsymbol{\lambda}^j = 1\right\}, \tag{7.14}$$

where $\mathbb{1}$ is a vector of ones.

Remark 9 *Notice that the approach in this paper is based on a commonly used "vertex enumeration approach". Its worst case complexity is exponential in horizon N of the feedback policy* (7.9)*, although independent on the length of the task duration T^j. We underline that this paper focuses on the fundamental properties of the controller design. Computationally tractability can be obtained as in any MPC scheme by using a different disturbance model or feedback parametrization.*

Remark 10 *We underline that the robust safe set in* (7.11) *may be constructed using just one predicted policy, i.e. setting $T^j = 0$. In this case, the proprieties of the proposed design still hold, and the computational complexity of constructing the convex safe set is reduced. Clearly, this approximation will likely shrink the domain of the proposed control policy.*

7.2.3 Robust Q-Function

The robust Q-function approximates the cost-to-go over the robust convex safe set and it is constructed iteratively as explained next. At iteration j we assume that we are given the robust Q-function $Q^{j-1}(\cdot)$ which maps each state $x \in \mathcal{CS}^{j-1}$ to the closed-loop cost, and we show how to construct a robust Q-function at the next iteration j. This recursion is initialized at iteration 0 setting the robust Q-function $Q^0(\cdot) = 0$ and the robust convex safe set $\mathcal{CS}^0 = \mathcal{O}$.

While in the nominal case from [61] the vertices of the convex safe set are a subset of the stored trajectory, the convex safe set from (7.11) may introduce additional vertices representing the worst case predicted realizations. For this reason, a cost-to-go associated with such predicted worst case realizations should be defined. In the following we defined the cost-to-go $J^j_{t|t}$ associated with the stored states $x^j_{t|t} = x^j_t$ and predicted cost-to-go $J^j_{k|t}$ associated with the predicted state $x^j_{k|t}$ at time k. In particular, after completion of the iteration j for $t \in \{0, \dots, T^j\}$, $k \in \{0, \dots, N-1\}$ and $i \in \{1, \dots, l^{k-1}\}$, we compute the cost-to-go as for the vertices $v^{j,i}_{k|t}$ of $\mathcal{CS}^j$ from $\mathbf{X}^j$ in (7.13) as

$$\begin{aligned} J^j_{k|t}\big(v^{j,i}_{k|t}\big) = \min_{\boldsymbol{\gamma}_k \geq 0} \quad & h\big(v^{j,i}_{k|t}, \pi^j_{k|t}(v^{j,i}_{k|t})\big) + \sum_{r=j}^{l^{k+j}} \gamma^r J^j_{k+1|t}(v^{j,r}_{k+1|t})), \\ \text{s.t.} \quad & \sum_{r=1}^{l^{k+1}} \gamma^r v^{j,r}_{k+1|t} = A v^{j,i}_{k|t} + B \pi^j_{k|t}(v^{j,i}_{k|t}) \\ & \sum_{r=1}^{l^{k+1}} \gamma^r = 1 \end{aligned} \tag{7.15}$$

for $\boldsymbol{\gamma}_k = [\gamma^1, \dots, \gamma^{l^{k+1}}]$ where l^{k+1} is the number of vertices $k+1$-steps robust reachable set $\mathcal{R}_{t \to t+k+1}(x^j_t)$. In the above recursion we set

$$J^j_{t+N|t}\big(v^{j,i}_{t+N|t}\big) = Q^{j-1}\big(v^{j,i}_{t+N|t}\big). \tag{7.16}$$

Basically, the cost-to-go $J^j_{k|t}(v^{j,i}_{k|t})$ at time k is computed summing up the running cost and the interpolated cost-to-go at the next predicted time $k+1$.

Given $Q^{j-1}(\cdot)$, the cost-to-go $J^i_{k|t}(\cdot)$ is computed for all $i \in \{0,\dots,j\}$, $t \in \{0,\dots,T^j\}$ and $k \in \{0,\dots,N\text{-}1\}$. Then, these cost values are collected in the following vector

$$\begin{aligned}\mathbf{J}^j = [\mathbf{J}^{j-1},\, & J^j_{0|0}\big(v^{j,1}_{0|0}\big),\dots,J^j_{N-1|0}\big(v^{j,l^{N-1}}_{N-1|0}\big),\dots\\ & J^j_{0|t}\big(v^{j,1}_{0|t}\big),\dots,J^j_{N-1|t}\big(v^{j,l^{N-1}}_{N-1|t}\big),\dots\\ & J^j_{0|T^j}\big(v^{j,1}_{N-1|T^j}\big),\dots,J^j_{N-1|T^j}\big(v^{j,l^{N-1}}_{N-1|T^j}\big)],\end{aligned}$$

where $\mathbf{J}^0 = [0,\dots,0]$ represents the cost-to-go associated with the vertices of $\mathcal{O}$. Finally, we define the Q-function at iteration j which interpolates the cost-to-go over the robust safe set,

$$Q^j(x) = \min_{\boldsymbol{\lambda}^j \in \Lambda^j(x)} \quad \mathbf{J}^j\boldsymbol{\lambda}^j, \tag{7.17}$$

where for the matrix $\mathbf{X}^j$ composed of $\mathrm{col}(\mathbf{X}^j)$ columns

$$\Lambda^j(x) = \Big\{\boldsymbol{\lambda}^j \in \mathbb{R}^{\mathrm{col}(\mathbf{X}^j)} : \boldsymbol{\lambda}^j \geq 0,\ \mathbf{X}^j\boldsymbol{\lambda}^j = x \text{ and } \mathbb{1}^\top\boldsymbol{\lambda}^j = 1\Big\} \tag{7.18}$$

collects the vectors $\boldsymbol{\lambda}^j$ which can be used to express x as a convex combination of the columns of $\mathbf{X}^j$.

7.2.4 Set of Safe Policies

At this point we have shown how to compute a robust invariant terminal set and a robust cost-to-go based on data collected at previous iterations. The last missing element needed for a MPC design is the feedback controller associated to the terminal set.

Here we show how to construct a set of safe policies $\mathcal{SP}^j$, which may be used to robustly constraint the evolution of system (7.1) into $\mathcal{CS}^j$, while satisfying state and input constraints (7.2). We begin by presenting an implicit parametrization of the set of policies $\mathcal{SP}^j$ which is amenable for optimization and it can be used to design a predictive controller that guarantees recursive constraint satisfaction. Afterwards, we define a safe policy $\kappa^{j,*}(\cdot) \in \mathcal{SP}^j$, which is able to complete the task from any state into the robust convex safe set $\mathcal{CS}^j$.

First, we define the matrix $\mathbf{U}^j$ collecting the inputs associated with the data stored in (7.13),

$$\begin{aligned}\mathbf{U}^j = [\mathbf{U}^{j-1},& \pi^j_{0|0}\big(v^{j,1}_{0|0}\big),\dots,\pi^j_{N-1|0}\big(v^{j,l^{N-1}}_{N-1|0}\big),\dots\\ & \pi^j_{0|t}\big(v^{j,1}_{0|t}\big),\dots,\pi^j_{N-1|t}\big(v^{j,l^{N-1}}_{N-1|t}\big),\dots\\ & \pi^j_{0|T^j}\big(v^{j,1}_{0|T^j}\big),\dots,\pi^j_{N-1|T^j}\big(v^{j,l^{N-1}}_{N-1|T^j}\big)]\end{aligned} \tag{7.19}$$

where the policies $\pi^j_{k|t}$ are defined in (7.5). In the above definition $\mathbf{U}^0 = [-Kv^1_o,\dots,-Kv^m_o]$, where the feedback gain K and the vertices v^i_o of $\mathcal{O}$ are defined in Assumption 10.

Now, we notice that by linearity of system (7.1), if a state $x \in \mathcal{CS}^j$ is expressed as a convex combination of the stored states $x = \mathbf{X}^j \boldsymbol{\lambda}^j$, then the input $u = \mathbf{U}^j \boldsymbol{\lambda}^j \in \mathcal{U}$ will keep the evolution of the system in $\mathcal{CS}^j$ for all disturbance realizations. More formally, given the set $\Lambda^j(\cdot)$ defined in (7.18), we have that $\forall x \in \mathcal{CS}^j \subseteq \mathcal{X}, \forall \boldsymbol{\lambda}^j \in \Lambda^j(x)$

$$\mathbf{U}^j \boldsymbol{\lambda}^j \in \mathcal{U} \text{ and } Ax + B\mathbf{U}^j \boldsymbol{\lambda}^j + w \in \mathcal{CS}^j, \forall w \in \mathcal{W}.$$

Therefore, the set of feedback policies $\kappa^j(\cdot) : \mathbb{R}^n \to \mathbb{R}^d$

$$\mathcal{SP}^j = \left\{ \kappa^j(\cdot) : \forall x \in \mathcal{CS}^j, \exists \boldsymbol{\lambda}^j \in \Lambda^j(x), \text{ such that } \kappa^j(x) = \mathbf{U}^j \boldsymbol{\lambda}^j \right\}, \tag{7.20}$$

guarantees that $\forall \kappa^j(\cdot) \in \mathcal{SP}^j$ the robust convex safe set $\mathcal{CS}^j$ is a robust positive invariant set for the closed-loop system $x_{t+1} = Ax_t + B\kappa^j(x_t) + w_t$. This statement is formalized by the following Proposition 6.

Proposition 6 *Let Assumptions 9-11 hold. Then, for all control policy $\kappa^j(\cdot) \in \mathcal{SP}^j$ and $\forall x \in \mathcal{CS}^j$ we have that*

$$Ax + B\kappa^j(x) + w \in \mathcal{CS}^j \subseteq \mathcal{X} \ \forall w \in \mathcal{W}$$

and $\kappa^j(x) \in \mathcal{U}$.

Proof *The proof can be found in Section 7.6.1 of the Appendix.*

Finally, we define the safe policy

$$\kappa^{j,*}(x) = \mathbf{U}^j \boldsymbol{\lambda}^{j,*}(x) \tag{7.21}$$

where $\boldsymbol{\lambda}^{j,*}(x)$ is the minimizer in (7.17). Basically, the above safe policy evaluated at x is given by the convex combination of stored inputs, for the multipliers $\boldsymbol{\lambda}^{j,*}(x)$ which define the robust Q-function at x. In the following propositions, we show that the Q-function is a Lyapunov function for the certainty equivalent closed-loop system (7.4) and (7.21). Furthermore, we show that the policy (7.21) in closed-loop with system (7.1) guarantees Input-to-State Stability (ISS).

Proposition 7 *Let Assumptions 9-11 hold. Consider the Q-function $Q^j(\cdot)$ in (7.17), we have that for all $x \in \mathcal{CS}^j$*

$$Q^j(x) \geq h(x, \kappa^{j,*}(x)) + Q^j(Ax + B\kappa^{j,*}(x)) \tag{7.22}$$

where $\kappa^{j,*}(\cdot)$ is the safe policy defined in (7.21).

Proof *The proof can be found in Section 7.6.2 of the Appendix.*

Proposition 8 *Consider the system (7.1) in closed-loop with the safe policy (7.21). Let Assumptions 9-11 hold and assume that $x_0 \in \mathcal{CS}^j$, then the closed-loop system (7.1) and (7.21) is Input to State Stable for the robust positive invariant set $\mathcal{O}$.*

Proof *The proof can be found in Section 7.6.3 of the Appendix.*

7.3 Control Design

This section introduces the iterative control design procedure. At the end of iteration $j-1$, we collect a data set of costs, inputs and states which are used to construct the robust convex safe set and robust Q-function at iteration $j-1$, as described in the Section 7.2. Finally, we exploit these quantities to design a robust Learning Model Predictive Controller (LMPC) for the jth iteration. The LMPC policy is able to safely execute the control task and it can be used to collect new closed-loop data to design the controller at the next iteration $j+1$.

7.3.1 LMPC Policy Synthesis

In this section, we introduce the LMPC policy. For more details on the control design choices we refer to the discussion in Section 7.3.2 and to the properties description in Section 7.4.

We define the following optimal control problem for the state $x_t^j \in \mathbb{R}^n$ and the parameter $N_t^j \in \mathbb{R}$,

$$
\begin{aligned}
C^{\text{LMPC},j}_{t\to t+N}(x_t^j, N_t^j) = \min_{\substack{\boldsymbol{M}_t^j,\\ \boldsymbol{\lambda}_t^j, \boldsymbol{g}_t^j}} \quad & \sum_{k=t}^{t+N-1} h(\bar{x}_{k|t}^j, u_{k|t}^j(\bar{x}_{k|t}^j)) + Q^{j-1}(\bar{x}_{t+N|t}^j) && \text{(7.23a)}\\
\text{s.t.} \quad & && \text{(7.23b)}\\
& \bar{x}_{k+1|t}^j = A\bar{x}_{k|t}^j + Bu_{k|t}^j(\bar{x}_{k|t}^j), && \text{(7.23c)}\\
& x_{k+1|t}^j = Ax_{k|t}^j + Bu_{k|t}^j(x_{k|t}^j) + w_{k|t}^j, && \text{(7.23d)}\\
& x_{t|t}^j = \bar{x}_{t|t}^j = x_t^j, && \text{(7.23e)}\\
& x_{k|t}^j \in \mathcal{X}, \quad u_{k|t}^j(x_{k|t}^j) \in \mathcal{U}, && \text{(7.23f)}\\
& x_{t+N|t}^j \in \ \mathcal{CS}^{j-1}, && \text{(7.23g)}\\
& \pi^{\text{d}}_{k|t}(x_{k|t}) = \textstyle\sum_{s=0}^{k-t-1} M_{ks|t}^j w_{s|t}^j + g_{k|t}^j && \text{(7.23h)}\\
& \kappa_{k|t}^{j-1}(x_{k|t}^j) = \mathbf{U}^{j-1}\boldsymbol{\lambda}_{k|t}^j && \text{(7.23i)}\\
& \boldsymbol{\lambda}_{k|t}^j \in \Lambda^{j-1}(x_{k|t}^j) && \text{(7.23j)}\\
& u_{i|t}^j(x_{i|t}^j) = \pi^{\text{d}}_{i|t}(x_{i|t}), \forall i \in \{t, ..., t+N_t^j-1\} && \text{(7.23k)}\\
& u_{i|t}^j(x_{i|t}^j) = \kappa_{i|t}^{j-1}(x_{i|t}^j), \forall i \in \{t+N_t^j, ..., t+N-1\} && \text{(7.23l)}\\
& \forall w_{k|t}^j \in \mathcal{W}, \ \forall k = \{t, ..., t+N-1\}
\end{aligned}
$$

where the optimization variables are

$$
\boldsymbol{M}_t^j = \begin{bmatrix} 0 & \cdots & \\ M_{21|t} & 0 & \cdots \\ M_{31|t} & M_{32|t} & \ddots \\ \vdots & & \end{bmatrix}, \ \boldsymbol{g}_t^j = \begin{bmatrix} g_{t|t}^j \\ \vdots \\ g_{t+N-1|t}^j, \end{bmatrix}
$$

$\boldsymbol{\lambda}_t^j = [\boldsymbol{\lambda}_{t|t}^j, \dots, \boldsymbol{\lambda}_{t|N-1|t}^j]$. Equations (7.23k)-(7.23l) and the parameter N_t^j describe the control policy which defines the evolution of the predicted nominal and uncertain trajectories in (7.23c)-(7.23d). In particular, for the first N_t^j predicted time steps the control policy $u_{k|t}^j(\cdot)$ equals the disturbance feedback policy (7.23h), and for the last $N - N_t^j$ predicted steps $u_{k|t}^j(\cdot)$ equals the safe feedback policy (7.23i)-(7.23j). Equations (7.23e)-(7.23f) represent input and state constraints which must be satisfied robustly for all disturbance realizations. Finally, the terminal constraint (7.23g) robustly forces $x_{t+N|t}$ into the robust control invariant set $\mathcal{CS}^{j-1}$.

The finite time optimal control problem (7.23) is used to define the LMPC algorithm described in Algorithm 1. Given the measured state x_t^j, Algorithm 1 solves $N+1$ instances of Problem (7.23) and it returns the control policy $\boldsymbol{u}_t^{j,*}(\cdot) = [u_{t|t}^{j,*}(\cdot), \dots, u_{t+N-1|t}^{j,*}(\cdot)]$ and the LMPC cost $J_{t\to t+N}^{\text{LMPC},j}(x_t^j)$. Then, we apply to system (7.1)

$$u_t^j = u_{t|t}^{j,*}(x_t). \tag{7.24}$$

Algorithm 1 is resolved at time $t+1$, based on the new state $x_{t+1|t+1} = x_{t+1}^j$, yielding a *moving* or *receding horizon* control strategy.

Algorithm 1: LMPC Algorithm

Given x_t^j
Set $N_t^{j,*} = \text{argmin}_{N_t^j \in \{0,\dots,N\}} C_{t\to t+N}^{\text{LMPC},j}(x_t^j, N_t^j)$
Let $\boldsymbol{u}_t^{j,*}(\cdot) = [u_{t|t}^{j,*}(\cdot), \dots, u_{t+N-1|t}^{j,*}(\cdot)]$ be the optimal solution to problem $C_{t\to t+N}^{\text{LMPC},j}(x_t^j, N_t^{j,*})$
Set $J_{t\to t+N}^{\text{LMPC},j}(x_t) = \min_{N_t^j \in \{0,\dots,N\}} C_{t\to t+N}^{\text{LMPC},j}(x_t^j, N_t^j)$
Return $\boldsymbol{u}_t^{j,*}(\cdot)$ and $J_{t\to t+N}^{\text{LMPC},j}(x_t^j)$

7.3.2 Design Choices

In standard robust MPC at each time step we a solve an optimal control problem over a fixed space of feedback policies. On the other hand, in Problem (7.23) the space of feedback policies changes as a function of the predicted time step k. Indeed, the predicted trajectory is computed using a disturbance feedback policy for $k \le N_t^j$ and a safe feedback policy (7.23i)-(7.23j) for $k > N_t^j$. In the following we discuss why this strategy allows us to guarantee recursive constraint satisfaction.

Recall that in predictive control recursive constraint satisfaction is ensured using a terminal constraint set. In particular, the terminal constraint set should be (robust) control invariant, for a feedback policy that can be used by the (robust) MPC to forecast the evolution of the system [10]. Notice that a disturbance feedback policy (or equivalently an affine state feedback policy [19]) may not be able to robustly constraint the evolution of the system into the terminal constraint set $\mathcal{CS}^j$. For this reason, in Problem (7.23) we used a time-varying feedback policy, which is defined by the parameter N_t^j, and in Algorithm 1 we

solved Problem (7.23) for different values of N_t^j. This strategy guarantees that the safe policy can be used to robustly constraint the evolution of the predicted system into the robust safe set $\mathcal{CS}^j$, and it is used in Theorem 17 to show that the LMPC (7.23) and (7.24) guarantees recursive feasibility.

Finally, we comment on the computational tractability of the proposed strategy. As already mentioned, Algorithm 1 solves $N+1$ instances of Problem (7.23) to forecast the evolution of the system using either the disturbance feedback policy or the safe policy from Section 7.2.4. We underline that these $N+1$ optimal control problems are independent and can be solved in parallel. Therefore, when parallel computing is available, the online computational complexity of the proposed strategy is independent on the controller horizon.

7.4 Properties

In this section we show that the LMPC policy (7.24) satisfies our design requirements from Section 7.1.1.

7.4.1 Recursive Feasibility

We show that if Problem (7.23) is feasible at time $t=0$ for some $N_0^j \in \{0,\dots,N\}$, then the LMPC policy (7.24) guarantees that state and input constraints are recursively satisfied. More precisely, we show that if $x_0^j \in \mathcal{C}^j$, where

$$\mathcal{C}^j = \{x \in \mathbb{R}^n : \exists N_0^j \in \{0,\dots,N\}, C_{0\to N}^{\text{LMPC},j}(x, N_0^j) < \infty\} \tag{7.25}$$

collects the states from which Problem (7.23) is feasible for some $N_0^j \in \{0,\dots,N\}$, then Problem (7.23) is feasible for all time $t \geq 1$ for some $N_t^j \in \{0,\dots,N\}$.

Theorem 17 *Consider the closed-loop system* (7.1) *and* (7.24). *Let Assumptions 9-11 hold and* $x_0^j \in \mathcal{C}^j$. *Then, for all time* $t \geq 0$ *the Problem* (7.23) *is feasible for some* $N_t^j \in \{0,\dots,N\}$, *and the closed-loop system* (7.1) *and* (7.24) *satisfies state and input constraints.*

Proof *Assume that at time t Problem* (7.23) *is feasible for some* $N_t^j \in \{0,\dots,N\}$. *At the next time* $t+1$, *by Proposition 6 we that, for* $\kappa^{j-1,*}(\cdot) \in \mathcal{SP}^{j-1}$, *the following candidate policy*

$$[u_{t+1|t}^{j,*}(\cdot),\dots,u_{t+N-1|t}^{j,*}(\cdot),\kappa^{j-1,*}(\cdot)] \tag{7.26}$$

is feasible for the Problem (7.23) *for some* $N_{t+1}^j \in \{0,\dots,N\}$.
By assumption we have that the Problem (7.23) *is feasible at time* $t=0$ *for some* $N_0^j \in \{0,\dots,N\}$. *Furthermore, we have shown that if Problem* (7.23) *is feasible for some* $N_t^j \in \{0,\dots,N\}$ *at time t, then Problem* (7.23) *is feasible for some* $N_{t+1}^j \in \{0,\dots,N\}$ *at* $t+1$. *Therefore by induction we conclude that for all time* $t \geq 0$ *Problem* (7.23) *is feasible for some* $N_t^j \in \{0,\dots,N\}$ *and the closed-loop system* (7.1) *and* (7.24) *satisfies state and input constraints* (7.2).

7.4.2 Input to State Stability (ISS)

In this section we show that the closed-loop system (7.1) and (7.24) is ISS with respect to $\mathcal{O}$. We recall that in standard MPC strategies the finite time optimal control problem can be reformulated as a parametric QP. This fact is used in [19] to show smoothness of the value function and then to prove ISS. In the proposed approach, the value function from Algorithm 1

$$J^{\text{LMPC},j}_{t\to t+N}(x_t) = \min_{N^j_t\in\{0,\dots,N\}} C^{\text{LMPC},j}_{t\to t+N}(x^j_t, N^j_t)$$

is not given by the solution to a parametric Quadratic Program (QP). Therefore, smoothness cannot be guaranteed, and the standard technique from [19] cannot be used to prove ISS. Instead, we introduce the standard definition of dissipative-form ISS-Lyapunov function for the robust invariant set $\mathcal{O}$ [17, 21] and we show that the cost of the LMPC $J^{\text{LMPC},j}_{t\to t+N}(x_t)$ is a ISS-Lyapunov function.

Definition 16 *A dissipative-form ISS-Lyapunov function for the closed-loop system* (7.1) *and* (7.24) *and the invariant set $\mathcal{O}$ is a function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ such that there exists $\alpha_1, \alpha_2, \alpha \in \mathcal{K}_\infty$ and $\sigma \in \mathcal{K}$ so that for all $x \in \mathbb{R}^n$ and $w \in \mathbb{R}^m$,*

$$\alpha_1(|x|_{\mathcal{O}}) \leq V(x) \leq \alpha_2(|x|_{\mathcal{O}}) \tag{7.27a}$$

$$V(Ax + B\pi^j(x) + w) - V(x) \leq -\alpha(|x|_{\mathcal{O}}) + \sigma(||w||). \tag{7.27b}$$

Notice that, as in [21], no assumptions on the continuity of $V(\cdot)$ are required. However (7.27a) implies that $V(\cdot)$ is continuous on the boundary of $\mathcal{O}$. The above definition can be used to show that the closed-loop system (7.1) and (7.24) is ISS with respect to the set invariant set $\mathcal{O}$, as described by the following proposition.

Proposition 9 *The following statements are equivalent:*

- *The closed-loop system* (7.1) *and* (7.24) *is ISS with respect to the robust invariant set $\mathcal{O}$.*
- *There exists a dissipative-form ISS-Lyapunov function $V(\cdot)$.*

Proof *The proof follows from [21, Theorem 2.3] substituting $|x|$ with $|x|_{\mathcal{O}}$. Note that we can replace $|x|$ with $|x|_{\mathcal{O}}$ as by* (7.27a) *we have that $V(x) = 0$ iff $|x|_{\mathcal{O}} = 0$.*

Proposition 10 *Let Assumptions 9-11 hold and $x^j_t \in \mathcal{C}^j$. We define closed-loop system dynamics*

$$f^j_t(x^j_t, w^j_t) = Ax_t + Bu^{j,*}_{t|t}(x^j_t) + w^j_t,$$

where $u^{j,*}_{t|t}(\cdot)$ is the optimal policy from Algorithm 1. Then there exists $L > 0$ such that

$$J^{LMPC,j}_{t+1\to t+1+N}(f^j_t(x^j_t, w^j_t)) - J^{LMPC,j}_{t\to t+N}(x^j_t) \leq -\alpha(|x^j_t|_{\mathcal{O}}) + L||w^j_t||,$$

$\forall t \geq 0$, $\forall w^j_t \in \mathcal{W}$ and $\alpha \in \mathcal{K}_\infty$.

Proof *The proof can be found in Section of the Appendix*

The above propositions allow us to prove that the closed-loop system (7.1) and (7.24) is ISS with respect to $\mathcal{O}$.

Theorem 18 *Consider the closed-loop system system* (7.1) *and* (7.24)*. Let Assumptions 9-11 hold and assume that* $x_0 \in \mathcal{C}^j$*, then the closed-loop system* (7.1) *and* (7.24) *is Input to State Stable (ISS) for the robust positive invariant set* $\mathcal{O}$*.*

Proof *First we show that the set* $\mathcal{O}$ *is robust positive invariant for the closed-loop system* (7.1) *and* (7.24)*. Assume that at time* t *of iteration* j *the state* $x_t^j \in \mathcal{O}$ *and recall that the disturbance feedback policy* (7.23h) *is equivalent to state feedback policy [19], then we have that the candidate policy*

$$[u_{t|t}^{j,*}(x) = -Kx, \dots, u_{t+N-1|t}^{j,*}(x) = -Kx]$$

is feasible at t *of the* j*th iteration for* $N_t^j = N$*. Now we notice that the cost associated with the above feasible policy is zero. Therefore, we have that* $u_t^j = u_{t|t}^{j,*}(x_t^j) = -Kx_t^j$*, which together with Assumption 10 implies that closed-loop system* $x_{t+1}^j = Ax_t^j + Bu_{t|t}^{j,*}(x_t^j) + w_t^j \in \mathcal{O}, \forall w_t^j \in \mathcal{W}$ *and that* $\mathcal{O}$ *is robust positive invariant for the closed-loop system* (7.1) *and* (7.24)*.*
We notice that LMPC cost from Algorithm 1 is time-invariant and we replace $J_{t\to t+N}^{LMPC,j}(\cdot)$ *with* $J_{0\to N}^{LMPC,j}(\cdot)$*. Furthermore, Assumption 11 and* (7.23a) *imply the existence of* $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ *such that* $\forall x_t \in \mathcal{C}^j$

$$\alpha_1(|x_t|_{\mathcal{O}}) \leq h(x_t, 0) \leq J_{0\to N}^{LMPC,j}(x_t) \leq \alpha_2(|x_t|_{\mathcal{O}}). \tag{7.28}$$

Finally, from Proposition 10, we have that $\forall x_t \in \mathcal{C}^j$

$$\begin{aligned} J_{0\to N}^{LMPC,j}(Ax_t + Bu_{t|t}^{j,*}(x_t) + w_t) - J_{0\to N}^{LMPC,j}(x_t) \\ \leq -\alpha(|x_t|_{\mathcal{O}}) + \sigma(||w_t||_2) \end{aligned}$$

and therefore $J_{0\to N}^{LMPC,j}(\cdot)$ *is a ISS-Lyapunov function and the closed-loop system* (7.1) *and* (7.24) *is Input to State Stable for the robust positive invariant set* $\mathcal{O}$*.*

7.4.3 Performance Bound

Finally, we show that whenever $x_0^j \in \mathcal{CS}^{j-1}$ the robust Q-function at iteration $j-1$ can be used to upper-bound the performance of the certainty equivalent system at the next jth iteration.

Theorem 19 *Consider the certainty equivalent system* (7.4) *in closed-loop with the LMPC* (7.23) *and* (7.24)*. Let Assumptions 9-11 hold and* $x_0^j \in \mathcal{CS}^{j-1} \subseteq \mathcal{C}^j$*, then we have that the*

iteration cost of the certainty equivalent closed-loop system (7.4) *and* (7.24) *is upper-bounded by the Q-function constructed at the previous iteration,*

$$J^j_{0\to\infty}(x^j_0) = \sum_{t=0}^{\infty} h\big(\bar{x}^j_t, u^{j,*}_{t|t}(\bar{x}^j_t)\big) \leq Q^{j-1}(x^j_0) \tag{7.29}$$

where $u^{j,*}_{t|t}(\cdot)$ *given by Algorithm 1.*

Proof *By Proposition 7 we have that*

$$\begin{aligned} Q^{j-1}(\bar{x}^j_0) &\geq h(\bar{x}^j_0, \kappa^{j-1,*}(\bar{x}^j_0)) + Q^{j-1}(\bar{x}^j_1) \\ &\geq h(\bar{x}^j_0, \kappa^{j-1,*}(\bar{x}^j_0)) + h(\bar{x}^j_1, \kappa^{j-1,*}(\bar{x}^j_1)) \\ &\qquad + Q^{j-1}(\bar{x}^j_2) \\ &\geq \sum_{k=0}^{N-1} h(\bar{x}^j_t, \kappa^{j-1,*}(\bar{x}^j_t)) + Q^{j-1}(\bar{x}^j_{N^j_0}) \\ &\geq J^{LMPC,j}_{0\to N}(\bar{x}^j_0), \end{aligned} \tag{7.30}$$

where the last inequality hold by the feasibility of safe policy from Section 7.2.4 for $x^j_0 \in \mathcal{CS}^{j-1} \subseteq \mathcal{C}^j$.
Now consider the LMPC cost at time t*, by Proposition 7 we have that*

$$\begin{aligned} J^{LMPC,j}_{t\to t+N}(\bar{x}^j_t) &= \sum_{k=t}^{t+N-1} h(x^{j,*}_{k|t}, u^{j,*}_{k|t}(x^{j,*}_{k|t})) + Q^{j-1}(x^{j,*}_{t+N|t}) \\ &\geq \sum_{k=t}^{t+N-1} h(x^{j,*}_{k|t}, u^{j,*}_{k|t}(x^{j,*}_{k|t})) + h\Big(x^{j,*}_{t+N|t}, \kappa^{j-1,*}\big(x^{j,*}_{t+N|t}\big)\Big) \\ &\qquad + Q^{j-1}\Big(Ax^{j,*}_{t+N|t} + B\kappa^{j-1,*}\big(x^{j,*}_{t+N|t}\big)\Big) \\ &= h(x^{j,*}_{t|t}, u^{j,*}_{t|t}(x^{j,*}_{t|t})) + \sum_{k=t+1}^{t+N-1} h(x^{j,*}_{k|t}, u^{j,*}_{k|t}(x^{j,*}_{k|t})) \\ &+ h\Big(x^{j,*}_{t+N|t}, \kappa^{j-1,*}\big(x^{j,*}_{t+N|t}\big)\Big) \\ &\qquad + Q^{j-1}\Big(Ax^{j,*}_{t+N|t} + B\kappa^{j-1,*}\big(x^{j,*}_{t+N|t}\big)\Big) \\ &\geq h(\bar{x}^j_t, u^j_t) + J^{LMPC,j}_{t+1\to t+1+N^j_{t+1}}(A\bar{x}^j_t + Bu^j_t), \end{aligned}$$

which implies that the LMPC cost is decreasing over the closed-loop trajectory of the certainty equivalent closed-loop system,

$$J^{LMPC,j}_{t+1\to t+1+N}(\bar{x}^j_{t+1}) - J^{LMPC,j}_{t\to t+N}(\bar{x}^j_t) \leq -h(\bar{x}^j_t, u^j_t).$$

By Theorem 18 and Definition 15 we have that the certainty equivalent system asymptotically converges to $\mathcal{O}$. Therefore, using the above equation recursively and the convergence of the certainty equivalent system to $\mathcal{O}$, we have that

$$\begin{aligned} J^{LMPC,j}_{0\to N}(\bar{x}^j_0) &\geq \sum_{k=0}^{\infty} h(\bar{x}^j_0, u^j_0) + \lim_{t\to\infty} J^{LMPC,j}_{t\to t+N}(\bar{x}^j_t) \\ &= \sum_{k=0}^{\infty} h(\bar{x}^j_0, u^j_0). \end{aligned}$$

Finally, from the above expression and equation (7.30) *we have that*

$$Q^{j-1}(x^t_0) \geq J^{LMPC,j}_{0\to N}(\bar{x}^j_0) \geq \sum_{k=0}^{\infty} h(\bar{x}^j_0, u^j_0) = J^j_{0\to\infty}(x^j_0).$$

7.5 Examples

We test the proposed controller on a system subject to bounded additive uncertainty. First, we show that the proposed strategy is able to improve the performance of a system executing an iterative task. Afterwards, we show that the proposed LMPC can be used to iteratively construct a robust convex safe set $\mathcal{CS}^j$, which is defined over progressively larger regions of the state space. Finally, we show that the data collected by the LMPC can be exploited to construct the safe policy $\kappa^{j,*}$ from Section 7.2.4. In particular, we show that this safe policy $\kappa^{j,*}$ robustly steers the uncertain system from any state into the robust safe set $\mathcal{CS}^j$ to the goal set $\mathcal{O}$.

We consider the following double integrator system

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t + w_t,$$

where $w_t \in \{w \in \mathbb{R}^2 : ||w||_\infty \leq 0.1\}$, subject to the following constraints $x_t \in \mathcal{X} = \{x \in \mathbb{R}^2 : ||x||_\infty \leq 10\}$ and $u_t \in \mathcal{U} = \{u \in \mathbb{R} : ||u||_\infty \leq 1\}$ for all time instant $t \geq 0$. Furthermore, we define the running cost $h(x, u) = 10|x|_\mathcal{O} + |u|_{K\mathcal{O}}$, which satisfies Assumption (11).

7.5.1 Iterative Task

We use the LMPC (7.23) and (7.24) to iteratively regulate the system from $x_0 = [5.656; 0]$ to the robust invariant set $\mathcal{O}$. We use a robust MPC to perform the 0th iteration and to construct the robust safe set $\mathcal{CS}^0$ and the robust Q-function $Q^0(\cdot)$, which are used to initialize the LMPC with $N = 3$.

We perform 4 iterations of the control task for the certainty equivalent system. At each jth iteration, we store the LMPC predicted policy and the closed-loop data in order to

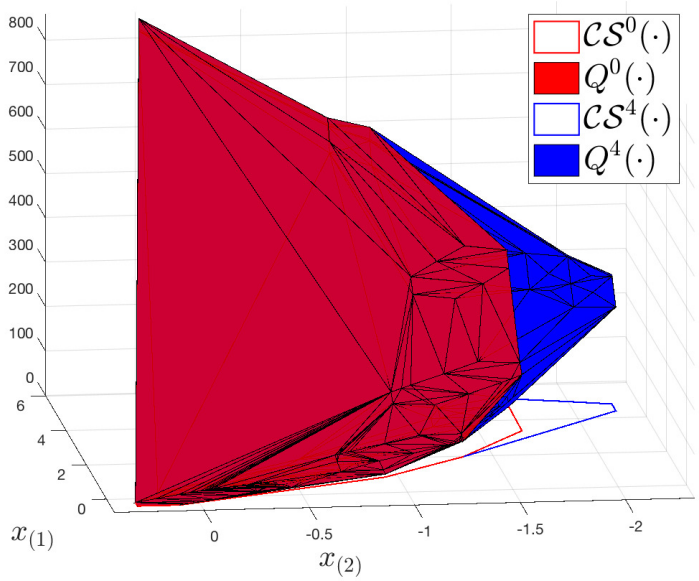


Figure 7.3: Comparison between the robust safe set and Q-function at the first and last iteration.

construct the robust safe set $\mathcal{CS}^j$ and the robust Q-function $Q^j(\cdot)$. Table 7.1 shows that the closed-loop cost of the certainty equivalent system decreases, until it converges to a steady state value after 4 iterations.

Table 7.1: Closed-loop cost $J^j_{0\to\infty}(x_0)$ for iteration $i \in \{0, \dots, 4\}$.

$i=0$	$i=1$	$i=2$	$i=3$	$i=4$
863.4245	827.9588	827.9380	827.9371	827.9371

Finally, in Figure 7.3 we compare the robust safe set and the robust Q-function at the first and last iteration. First, we notice that the robust safe set, which represents the domain of the Q-function, is enlarged. Furthermore, we confirm that $Q^j(\cdot)$ is non-increasing (i.e. $Q^1(x) \leq Q^5(x), \forall x \in \mathcal{CS}^5$) and therefore it guarantees better bounds on the performance of certainty equivalent closed-loop system (7.4) and (7.24), as shown in Theorem 19.

7.5.2 LMPC Domain Enlargement

We show that the domain of the LMPC policy may be iteratively enlarged. At each iteration, we simulate the uncertain closed-loop system (7.1) and (7.24) and we store both the closed-loop data and the predicted LMPC policy from Algorithm 1. These stored data are used to construct the robust safe set $\mathcal{CS}^j$ and robust Q-function as described in Section 7.2.

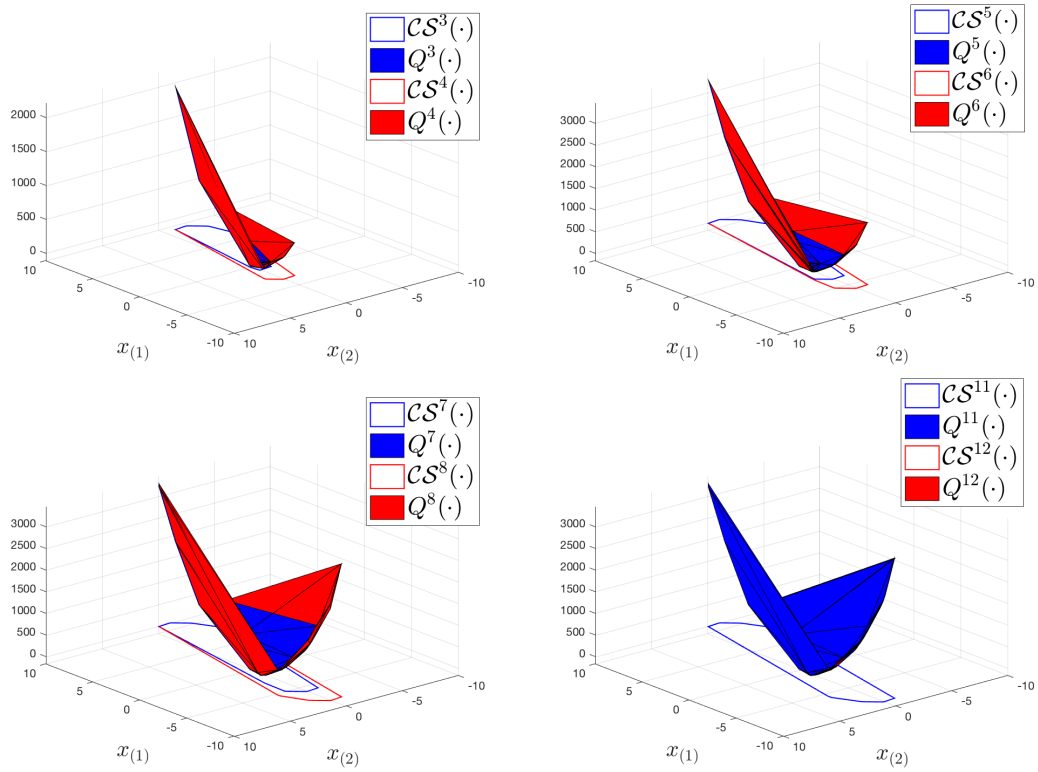


Figure 7.4: Evolution of the robust Q-function Q^j through the iterations. Notice that $Q^j(\cdot)$ (in blue) is lower-bounded by $Q^{j+1}(\cdot)$ (in red) for all $i \in \{3, 5, 7, 11\}$, unitl convergence is reached and $Q^{11}(\cdot) = Q^{12}(\cdot)$.

Notice that by definition (7.11) $\mathcal{CS}^j \subseteq \mathcal{CS}^{j+1}$ therefore the set of states which can be steered to $\mathcal{O}$ by the LMPC (7.23) and (7.24) does not shrink (i.e. $\mathcal{C}^j \subseteq \mathcal{C}^{j+1}$). At each jth iteration, we compute the initial condition x_0^j as the furthest point along the positive or negative x-axis from which the LMPC is feasible. More formally, at each the jth iteration the initial state x_0^j is determined solving the following convex optimization problem

$$x_0^j = \arg \min_{\substack{x \in \mathcal{C}^j \\ bx=0}} a^j x$$

where the row vectors $a^j = [(-1)^j, 0] \in \mathbb{R}^2$, $b = [0, 1] \in \mathbb{R}^2$ and $\mathcal{C}^j$ is defined in (7.25).

Figure 7.5 shows that the robust convex safe set $\mathcal{CS}^j$ grows at each iteration until it converges to a set which saturates the state constraints. We underline that the closed-loop data used to enlarge $\mathcal{CS}^j$ are generated by the LMPC, which steers the system to regions of the state space associated with low cost values. In other words, the growth of the robust safe set is cost-driven. More importantly, the iterative enlargement of the $\mathcal{CS}^j$ is performed

safely. Indeed, the LMPC guarantees robust state and input constraints satisfaction at each iteration.

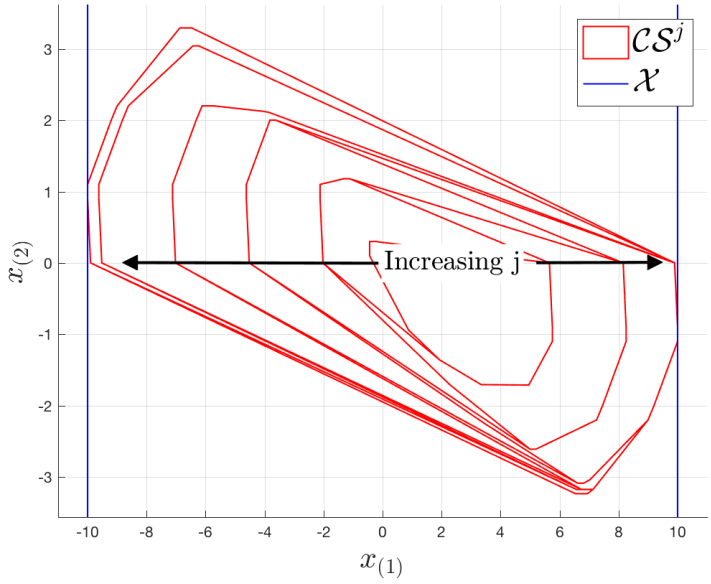


Figure 7.5: Evolution of the robust safe set through the iterations

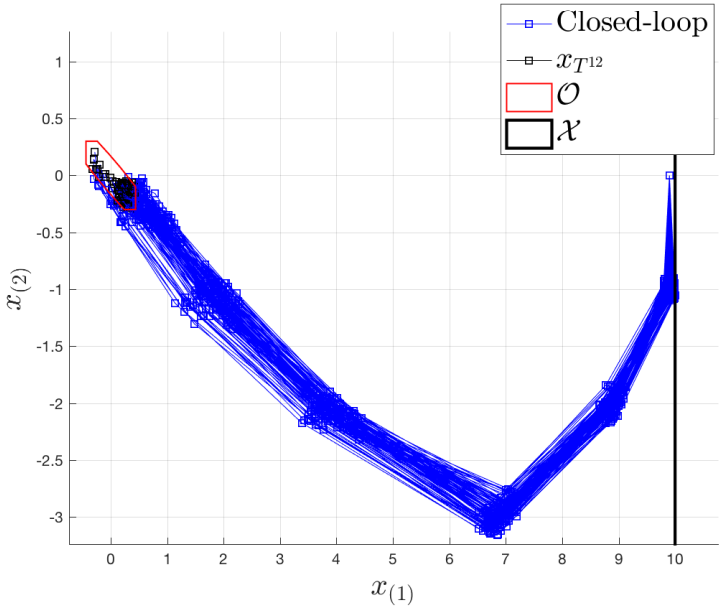


Figure 7.6: Closed-loop trajectories for different disturbance realizations.

Figure 7.4 shows the growth of the Q-function $Q^j(\cdot)$, which is non-increasing through the iterations. It is important to underline that $Q^j(\cdot)$ is piece-wise affine as it is the solution to a parametric LP [10]. Furthermore, we notice that $Q^j(\cdot)$, which upper-bounds the closed-loop cost of the disturbance-fee system, resembles a quadratic function. This result makes sense as the optimal value function for this problem is piece-wise quadratic [10]. Finally, Figure 7.6 shows 100 Monte Carlo simulations of the closed-loop system for the 12th iteration. We notice that the closed-loop trajectories satisfy state constraints and converge to the goal set $\mathcal{O}$, regardless of the disturbance realization.

7.5.3 Exploiting the safe policy

Finally, we use the stored data from the previous Section 7.2.4 to construct the safe policy (7.21). We tested this policy for 1000 Monte Carlo simulations, where we randomly sampled the initial condition x_0 from the robust convex safe set $\mathcal{CS}^{12}$. We confirm that, for all initial conditions $x_0 \in \mathcal{CS}^{12}$ and disturbance realization, the safe policy (7.21) steered the system to the goal set $\mathcal{O}$, as shown in Figure 7.7.

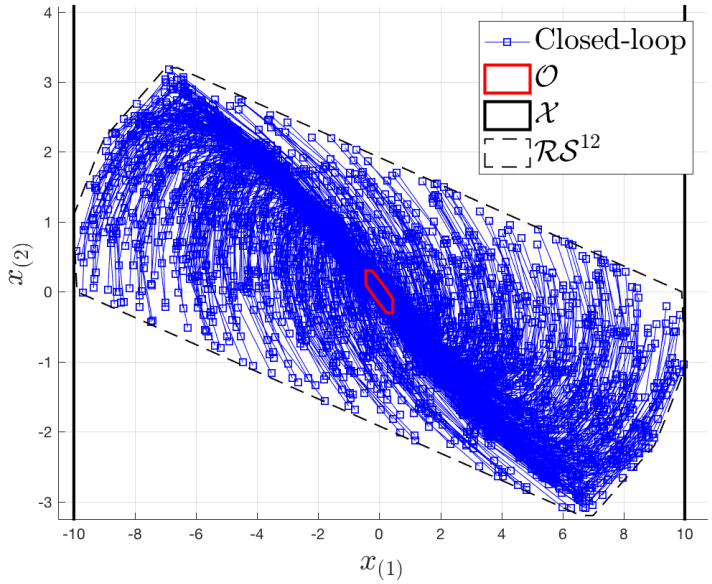


Figure 7.7: Closed-loop trajectories for different disturbance realizations and initial conditions.

Finally, we compare the performance of the uncertain system (7.1) in closed-loop with the safe policy (7.21) and the LMPC (7.23) and (7.24). In particular, we simulated both the closed-loop system (7.1) and (7.21) and the closed-loop system (7.1) and (7.24) for the same random initial condition $x_0 \in \mathcal{CS}^{12}$ and disturbance realization. As reported in Table 7.2, on average it takes $\sim$ 5ms to evaluate the safe policy (7.21) and $\sim$ 4.6s to evaluate the LMPC

policy (7.24). This result is expected as the safe policy (7.21) is evaluated solving a LP and the LMPC policy (7.24) solving $N+1=4$ QPs. On the other hand, it is interesting to notice that the closed-loop cost associated with safe policy (7.21) is on average $\sim 3\%$ higher than the cost associated with the LMPC policy (7.24), as shown in Table 7.2. This result suggests that, in applications where the computational power is not always available, one can first use the proposed LMPC to iteratively construct a large robust convex safe set $\mathcal{CS}^j$ and robust Q-function Q^j. Afterwards, these quantities can be exploited to synthesis a safe control policy, which at the cost of slickly worse performance is able reduce the computational burden.

Table 7.2: Performance of the LMPC policy (7.24) and the safe policy (7.21) in closed-loop with the uncertain system (7.1).

	Average solver time	Average closed-loop cost
LMPC	4.6s	77.1
Safe Policy	5ms	80

7.6 Appendix

7.6.1 Proof of Proposition 6

The proof follows from linearity of system (7.1) and convexity of the constraint set (7.2). By Assumption 13 we have that for all $v_t^i(x_t^j)$ collected in the columns of the matrix $\mathbf{X}^j$ in (7.13)

$$Av_t^i(x_t^j) + B\pi_{k|t}^j(v_t^i(x_t^j)) + w \in \mathcal{CS}^j \subseteq \mathcal{X},\ \forall w \in \mathcal{W}.$$

The above equation implies that $\forall x \in \mathcal{CS}^j$ and $\forall \boldsymbol{\lambda}^j \in \Lambda(x)$

$$A\mathbf{X}^j\boldsymbol{\lambda}^j + B\mathbf{U}^j\boldsymbol{\lambda}^j + w \in \mathcal{CS}^j, \forall w \in \mathcal{W}.$$

By definition $\forall \boldsymbol{\lambda}^j \in \Lambda(x)$ and $\kappa^j(\cdot) \in \mathcal{SP}^j$, we have that $x = \mathbf{X}^j\boldsymbol{\lambda}^j$ and $\kappa^j(x) = \mathbf{U}^j\boldsymbol{\lambda}^j$. Consequently, from the above equation we have that $\forall x \in \mathcal{CS}^j$ and $\kappa^j(\cdot) \in \mathcal{SP}^j$

$$Ax + B\kappa^j(x) + w \in \mathcal{CS}^j, \forall w \in \mathcal{W}.$$

■

7.6.2 Proof of Proposition 7

Recall that we initialized $\mathcal{CS}^0 = \mathcal{O}$ and $Q^0(x) = 0\ \forall x \in \mathcal{CS}^0$, then we trivially have that

$$Q^0(x) \geq h(x, \kappa^{0,*}(x)) + Q^0(Ax + B\kappa^{0,*}(x)), \forall x \in \mathcal{CS}^0.$$

Now, we show that $\forall j \geq 1$ and $\forall x \in \mathcal{CS}^{j-1}$

$$Q^j(x) \geq h(x, \kappa^{j,*}(x)) + Q^j(Ax + B\kappa^{j,*}(x)).$$

Let $x \in \mathcal{CS}^j$ then we have

$$Q^j(x) = (\mathbf{J}^j)^\top \boldsymbol{\lambda}^{*,j}. \tag{7.31}$$

Now notice that by definitions (7.15)-(7.16) each element of $\mathbf{J}^j$ can be written as

$$\begin{aligned} J^j_{k|t}\big(v^i_k(x^j_t)\big) &= h\big(v^i_k(x^j_t), \pi^j_{k|t}(v^i_k(x^j_t))\big) \sum_{r=1}^{l^{k+1}} \gamma^{r,*} J^j_{k+1|t}(v^r_{k+1}(x^j_t)) \\ &\geq h\big(v^i_k(x^j_t), \pi^j_{k|t}(v^i_k(x^j_t))\big) + Q^j\big(Av^i_k(x^j_t) + B\pi^j_{k|t}(v^i_k(x^j_t))\big). \end{aligned} \tag{7.32}$$

Convexity of $h(\cdot,\cdot), Q^j(\cdot)$ and Equation (7.32) imply that

$$\begin{aligned} Q^j(x) = \mathbf{J}^j\boldsymbol{\lambda}^{*,j} &\geq h\left(\begin{bmatrix} v^0_0(x^0_0) \\ \vdots \\ v^i_k(x^j_t) \\ \vdots \\ v^{l^{N-1}}_{N-1}(x^j_{T^j}) \end{bmatrix} \boldsymbol{\lambda}^{j,*}, \begin{bmatrix} \pi^0_{0|0}(v^0_0(x^0_0)) \\ \vdots \\ \pi^j_{k|t}(v^i_k(x^j_t)) \\ \vdots \\ \pi^j_{N-1|T^j}(v^{l^{N-1}}_{N-1}(x^j_{T^j})) \end{bmatrix} \boldsymbol{\lambda}^{j,*} \right) \\ &\quad + \begin{bmatrix} Q^j(Av^1_0(x^0_0) + B\pi^0_{0|0}\big(v^1_0(x^0_0)\big) \\ \vdots \\ Q^j(Av^i_k(x^j_t) + B\pi^j_{k|t}\big(v^i_k(x^j_t)\big) \\ \vdots \\ Q^j(Av^{l^N}_N(x^j_{T^j}) + B\pi^j_{N-1|T^j}\big(v^{l^{N-1}}_{N-1}(x^j_{T^j})\big) \end{bmatrix} \boldsymbol{\lambda}^{*,j} \\ &\geq h\big(\mathbf{X}^j\boldsymbol{\lambda}^{j,*}, \mathbf{U}^j\boldsymbol{\lambda}^{j,*}\big) + Q^j(\mathbf{X}^j\tilde{\boldsymbol{\lambda}}^{j,*}), \end{aligned} \tag{7.33}$$

for some $\tilde{\boldsymbol{\lambda}}^{j,*}$ such that

$$\mathbf{X}^j\tilde{\boldsymbol{\lambda}}^{j,*} = \begin{bmatrix} Av^1_0(x^0_0) + B\pi^0_{0|0}\big(v^1_0(x^0_0)\big) \\ \vdots \\ Av^i_k(x^j_t) + B\pi^j_{k|t}\big(v^i_k(x^j_t)\big) \\ \vdots \\ Av^{l^{N-1}}_{N-1}(x^j_{T^j}) + B\pi^{j-1}_{N-1|T^j}\big(v^{l^{N-1}}_{N-1}(x^j_{T^j})\big) \end{bmatrix} \boldsymbol{\lambda}^{j,*}.$$

The above equation implies that

$$\begin{aligned} Q^j(x) &\geq h\big(\mathbf{X}^j\boldsymbol{\lambda}^{j,*}, \mathbf{U}^j\boldsymbol{\lambda}^{j,*}\big) + Q^j(\mathbf{X}^j\tilde{\boldsymbol{\lambda}}^{j,*}) \\ &\geq h(\mathbf{X}^j\boldsymbol{\lambda}^{j,*}, \mathbf{U}^j\boldsymbol{\lambda}^{j,*}) + Q^j(A\mathbf{X}^j\boldsymbol{\lambda}^{j,*} + B\mathbf{U}^j\boldsymbol{\lambda}^{j,*}) \end{aligned}$$

Finally, we notice that by definition (7.21) $\kappa^{j,*}(x) = \mathbf{U}^j \boldsymbol{\lambda}^{j,*}$, therefore the above equation can be rewritten as

$$Q^j(x) \geq h(x, \kappa^{j,*}(x)) + Q^j(Ax + B\kappa^{j,*}(x)).$$

■

7.6.3 Proof of Proposition 8

First we show that the set $\mathcal{O}$ is robust positive invariant for the closed-loop system (7.1) and (7.21). Assume that at time t of iteration j the state $x_t^j \in \mathcal{O}$, be definitions (7.13), (7.15), (7.17) and (7.21) we have that $Q^j(x_t^j) = 0$ and $\kappa^{j,*}(x_t^j) = -Kx_t^j$. Therefore, by Assumption 10 we have that closed-loop system $x_{t+1}^j = Ax_t^j + B\kappa^{j,*}(x_t^j) + w_t^j \in \mathcal{O}, \forall w_t^j \in \mathcal{W}$ and that $\mathcal{O}$ is robust positive invariant for the closed-loop system (7.1)-(7.21).
We notice that Assumption 11 and (7.23a) imply the existence of $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ such that $\forall x_t \in \mathcal{CS}^j$

$$\alpha_1(|x_t|_{\mathcal{O}}) \leq h(x_t, 0) \leq Q^j(x_t) \leq \alpha_2(|x_t|_{\mathcal{O}}). \tag{7.34}$$

Now we notice that $Q^j(\cdot)$ is Lipschitz as it is the solution to a parametric LP [10]. Finally, from Proposition 10 and Lipschitz continuity of $Q^j(\cdot)$ for a Lipschitz constant L, we have that $\forall x_t \in \mathcal{CS}^j$

$$\begin{aligned} Q^j(Ax_t + B\kappa^{j,*}(x_t) + w_t) - Q^j(x_t) &= Q^j(Ax_t + B\kappa^{j,*}(x_t) + w_t) - Q^j(Ax_t + B\kappa^{j,*}(x_t)) \\ &\quad + Q^j(Ax_t + B\kappa^{j,*}(x_t)) - Q^j(x_t) \\ &\leq L||w_t||_2 + Q^j(Ax_t + B\kappa^{j,*}(x_t)) - Q^j(x_t) \\ &\leq L||w_t||_2 - h(x_t, \kappa^{j,*}(x_t)). \end{aligned}$$

Therefore $Q^j(\cdot)$ is a ISS-Lyapunov function according with Definition 15 and by Proposition 9 the closed-loop system (7.1) and (7.21) is Input to State Stable for the robust positive invariant set $\mathcal{O}$. ■

7.6.4 Proof of Proposition 10

By assumption $x_t^j \in \mathcal{C}^j$, therefore by Theorem 17 Problem (7.23) is feasible at time t for some $N_t^j \in \{0, \dots, N\}$ and the LMPC Algorithm 1 returns the optimal policy $\mathbf{u}_t^{j,*}(\cdot)$ and optimal cost $J_{t\to t+N}^{\text{LMPC},j}(x_t^j)$. At time t, let

$$[\bar{x}_{t|t}^{j,*}, \dots, \bar{x}_{t+N|t}^{j,*}]$$

the optimal trajectory of the certainty equivalent system associated with the optimal policy $\mathbf{u}_t^{j,*}(\cdot)$. Then, the LMPC cost from Algorithm 1 at time t can be written as

$$\begin{aligned} J_{t\to t+N}^{\text{LMPC},j}(x_t^j) &= \sum_{k=t}^{t+N-1} h(\bar{x}_{k|t}^{j,*}, u_{k|t}^{j,*}(\bar{x}_{k|t}^{j,*})) + Q^{j-1}(\bar{x}_{t+N|t}^{j,*}) \\ &= h(\bar{x}_{t|t}^{j,*}, u_{t|t}^{j,*}(\bar{x}_{k|t}^{j,*})) + p(\bar{x}_{t+1|t}^{j,*}) \end{aligned} \tag{7.35}$$

where the function $p(\bar{x}^{j,*}_{t+1|t})$, which represents the total cost from time $t+1$ to time $t+N$ for the optimal policy $\mathbf{u}^{j,*}_t(\cdot)$, is Lipschitz as it is composed of summation and composition of Lipschitz functions. Now we notice that by feasibility of (7.26) the cost of the LMPC at time $t+1$ can be upper-bounded,

$$\begin{aligned} J^{\text{LMPC},j}_{t+1\to t+1+N}(f(x_t,w_t)) \leq & \sum_{k=t+1}^{t+N-1} h\big(\bar{x}^j_{k|t+1}, u^{*,j}_{k|t}(\bar{x}^j_{k|t+1})\big) + h\big(\bar{x}^j_{t+N|t+1}, \kappa^{j-1,*}(\bar{x}^j_{t+N|t+1})\big) \\ & + Q^{j-1}\big(A\bar{x}^j_{t+N|t+1} + B\kappa^{j-1,*}(\bar{x}^j_{t+N|t+1})\big) \\ & \leq \sum_{k=t+1}^{t+N-1} h\big(\bar{x}^j_{k|t+1}, u^{*,j}_{k|t}(\bar{x}^j_{k|t+1})\big) + Q^{j-1}\big(\bar{x}^j_{t+N|t+1}\big) \\ & = p(\bar{x}^j_{t+1|t+1}) \end{aligned} \tag{7.36}$$

where the last inequality follows from Proposition 7 and the feasible nominal trajectory is given by

$$\bar{x}^j_{k|t+1} = A^{k-t-1}(Ax_t + Bu^{*,j}_{t|t}(x_t) + w_t) + \sum_{i=t+1}^{k-1} A^{k-1-i} B u^{*,j}_{i|t}(\bar{x}^j_{k|t+1})$$

for $k = \{t+2, \dots, t+N\}$. Finally, we notice that

$$\bar{x}^j_{t+1|t+1} = Ax^j_t + Bu^{*,j}_{t|t}(x^j_t) + w^j_t = \bar{x}^{j,*}_{t+1|t} + w^j_t, \tag{7.37}$$

$\forall w^j_t \in \mathcal{W}$. Therefore, from the Lipschitz continuity of $p(\cdot)$, (7.35), (7.36) and (7.37) we have that

$$\begin{aligned} J^{\text{LMPC},j}_{t+1\to t+1+N}(f(x^j_t,w^j_t)) - J^{\text{LMPC},j}_{t\to t+N}(x^j_t) &= p(\bar{x}^{j,*}_{t+1|t} + w_t) - p(\bar{x}^{j,*}_{t+1|t}) - h(\bar{x}^{*,j}_{t|t}, u^{*,j}_{t|t}(x^j_t)) \\ &\leq -h(\bar{x}^{*,j}_{t|t}, u^{*,j}_{t|t}(x^j_t)) + L||w^j_t|| \\ &\leq -h(\bar{x}^{*,j}_{t|t}, 0) + L||w^j_t|| \\ &\leq -\alpha^l_x(x^j_t) + L||w^j_t||, \end{aligned} \tag{7.38}$$

$\forall w^j_t \in \mathcal{W}$, where the last inequality holds by Assumption 11. ■

Chapter 8

Certainty Equivalent Learning Model Predictive Control

In the previous chapters, we presented two robust LMPC design strategies. The computational burden associated with these strategies increases with the respect to the nominal case, as the controller forecasts the evolution of the system using feedback policies. In this chapter, we first introduce a decoupling strategy, which allows us to represent the uncertain system as the summation of a certainty equivalent and an error dynamics. Then, we propose a robust LMPC policy which exploits the certainty equivalent system to predict a nominal trajectory and the error dynamics to guarantee robust constraint satisfaction. The main advantage of proposed strategy is that the on-line computational cost does not increase with respect to the nominal case. Nonetheless, we show that robust constraint satisfaction and performance improvement properties for the closed-loop system.

8.1 Problem Statement

In this section, we illustrate the linear change of coordinates that allows us to represent the uncertain system as the summation of a certainty equivalent and an error dynamics. Afterwards, we introduce the robustified infinite horizon optimal control problem, which is used to define a feedback policy that guarantees robust constraint satisfaction.

8.1.1 The Certainty Equivalent System

Consider the uncertain linear time-invariant systems

$$x_{t+1} = Ax_t + Bu_t + w_t \tag{8.1}$$

where at time t the state $x_t \in \mathbb{R}^n$, the input $u_t \in \mathbb{R}^{n_u}$, and the matrices A and B are known. At each time step t, the system is affected by a random disturbance $w_t \in \mathcal{W}$, where $\mathcal{W}$ is

assumed to be a known compact polytope that contains the origin. We consider the state and input constraints

$$x_t \in \mathcal{X} \text{ and } u_t \in \mathcal{U} \tag{8.2}$$

which must be satisfied for all uncertainty realizations $w_t \in \mathcal{W}$ and for all $t \geq 0$..

In the following, we show that the uncertain system 8.1 may be decoupled into a nominal and uncertain error dynamics [81, 82]. First, we introduce the following certainty equivalent system and associated feedback policy

$$\begin{aligned} \bar{x}_{t+1} &= A\bar{x}_t + B\bar{u}_t \\ \pi(x_t) &= -K(x_t - \bar{x}_t) + \bar{u}_t \end{aligned} \tag{8.3}$$

where x_t is the state of system (8.1). The above certainty equivalent dynamics allows us to define the error at time t as

$$e_t = x_t - \bar{x}_t. \tag{8.4}$$

Finally, using (8.3) and (8.4), we rewrite (8.1) as

$$\begin{aligned} \bar{x}_{t+1} &= A\bar{x}_t + B\bar{u}_t \\ e_{t+1} &= (A - BK)e_t + w_t. \end{aligned} \tag{8.5}$$

Next, we assume that we are given a robust invaraint set $\mathcal{E}$ for the error dynamics and we present the constraint robustification strategy.

Assumption 15 *The set $\mathcal{E}$ is a robust positive invariant set for the error dynamics,*

$$\forall e \in \mathcal{E}, (A - BK)e + w \in \mathcal{E}, \forall w \in \mathcal{W},$$

and $\mathcal{U} \supseteq -K\mathcal{E} = \{-Ke \in \mathbb{R}^d : e \in \mathcal{E}\}$.

The above assumption implies that if e_0 belongs to some given set $\mathcal{E}$, then e_t belongs to the set $\mathcal{E}$ for all $w_t \in \mathcal{W}$ and $t \geq 0$. Moreover, from (8.5), it follows that $x_t = \bar{x}_t + e_t \in \bar{x}_t \oplus \mathcal{E}$ and $u_t = -Ke_t + \bar{u}_t \in \{-K\mathcal{E}\} \oplus \bar{u}_t$. This fact allows us to robustly enforce state and input constraints on x_t and u_t through the following constraints on the certainty equivalent system

$$\begin{aligned} \bar{x}_t &\in \bar{\mathcal{X}} = \mathcal{X} \ominus \mathcal{E} \\ \bar{u}_t &\in \bar{\mathcal{U}} = \mathcal{U} \ominus \{-K\mathcal{E}\}, \forall t \geq 0. \end{aligned} \tag{8.6}$$

Indeed, if $e_0 \in \mathcal{E}$ and (8.6) is satisfied we have that

$$\begin{aligned} &x_t = \bar{x}_t + e_t \in \bar{\mathcal{X}} \oplus \mathcal{E} = \mathcal{X}, \\ &u_t = -Ke_t + \bar{u}_t \in \{-K\mathcal{E}\} \oplus \bar{\mathcal{U}} = \mathcal{U}, \ \forall t \geq 0, \forall w_t \in \mathcal{W}. \end{aligned} \tag{8.7}$$

8.1.2 Robustified Infinite Horizon Optimal Control Problem

In this section, we introduce the robustified infinite horizon optimal control problem. In particular, we exploit the linear change of coordinates presented in the previous section to reformulate the robust constraints as a deterministic one. Finally, we show that the optimal solution to the robustified problem defines a feedback policy which guarantees robust constraint satisfaction.

Our goal is to synthesize a feedback policy which guarantees recursive robust constraint satisfaction for the closed-loop system. The key idea is to exploit invariance of the set $\mathcal{E}$ and the decoupling strategy presented in the previous section. In particular, we define the following infinite time optimal control problem

$$\begin{aligned} J^*_{0\to\infty}(x_S) = \min_{\bar{x}_0,\bar{u}_0,\bar{u}_1,\dots} \quad & \sum_{t=0}^{\infty} h(\bar{x}_t,\bar{u}_t) \\ \text{s.t.} \quad & \bar{x}_{t+1} = A\bar{x}_t + B\bar{u}_t, \forall t \in \{0,1,\dots\} \\ & \bar{x}_t \in \bar{\mathcal{X}}, \bar{u}_t \in \bar{\mathcal{U}}, \forall t \in \{0,1,\dots\} \\ & x_S - x_0 \in \mathcal{E}. \end{aligned} \tag{8.8}$$

Let $[\bar{x}^*_0, \bar{u}^*_0, \bar{u}^*_1, \dots]$ be the optimal optimal solution to (8.8) and $[x^*_0, \bar{x}^*_1, \dots]$ the associated certainty equivalent closed-loop trajectory. Then it follows from 8.6 that the feedback policy

$$\pi(x_t) = -K(x_t - \bar{x}^*_t) + \bar{u}^*_t \tag{8.9}$$

guarantees robust state and input constraint satisfaction.

In the following, we introduce a certainty LMPC strategy to iteratively compute a solution to the above infinite time optimal control problem. We show that the certainty LMPC policy guarantees iterative performance improvement and recursive robust constraint satisfaction. The main advantage of the strategy presented in this chapter is that the on-line computational cost does not increase with respect to the nominal case.

8.2 Controller Design

This section describes the controller design. First, we introduce the definition of safe set and Q-function for the certainty equivalent system (8.3), which follows from Chapter 4. Afterwards, we present the controller design.

8.2.1 Convex Safe Set

In this section, we define the safe set for the certainty equivalent system. At iteration j, let the vectors

$$\bar{\mathbf{u}}^j = [\bar{u}^j_0,\ \bar{u}^j_1,\ ...,\ \bar{u}^j_t,\ ...], \tag{8.10a}$$

$$\bar{\mathbf{x}}^j = [\bar{x}^j_0,\ \bar{x}^j_1,\ ...,\ \bar{x}^j_t,\ ...], \tag{8.10b}$$

denote the input and state of certainty equivalent system (8.3). To exploit the iterative nature of the control design, we define the *sampled Safe Set* $\mathcal{SS}^j$ at iteration j as

$$\mathcal{SS}^j = \left\{ \bigcup_{i \in M^j} \bigcup_{t=0}^{\infty} \bar{x}_t^i \right\}, \tag{8.11}$$

where $M^j \subseteq \{0, \dots, j\}$ is the set of indices associated with successful iterations, i.e.,

$$M^j = \left\{ k \in [0, j] : \lim_{t \to \infty} \bar{x}_t^k = 0 \right\}.$$

In other words, $\mathcal{SS}^j$ is the collection of all nominal state trajectories up to iteration j that have converged to the origin. Finally, we define the *convex Safe Set* as

$$\mathcal{CS}^j = \text{conv}(\mathcal{SS}^j). \tag{8.12}$$

8.2.2 Q-function

At time t of the jth iteration, the cost-to-go associated with the closed loop trajectory (8.10b) and input sequence (8.10a) is defined as

$$J_{t \to \infty}^j(\bar{x}_t^j) = \sum_{k=0}^{\infty} h(\bar{x}_{t+k}^j, \bar{u}_{t+k}^j),$$

where $h(\cdot, \cdot)$ is the stage cost of problem (8.8). Note that $J_{0 \to \infty}^j(s_0^j)$ quantifies the controller performance at the jth iteration.

We are now in place to define the barycentric function [63]

$$\begin{aligned} Q^j(\bar{x}) := \min_{\lambda_t \geq 0, \forall t \in [0,\infty)} \quad & \sum_{i=0}^{j} \sum_{t=0}^{\infty} \lambda_t^i J_{t \to \infty}^i(\bar{x}_t^i) \\ \text{s.t.} \quad & \sum_{i=0}^{j} \sum_{t=0}^{\infty} \lambda_t^i \bar{x}_t^i = \bar{x}, \sum_{i=0}^{j} \sum_{t=0}^{\infty} \lambda_t^i = 1, \end{aligned}$$

where $\bar{x}_t^i$ is the nominal state at time t of the i-th iteration, as defined in (8.10b). The function $Q^j(\cdot)$ assigns to every point in $\mathcal{CS}^j$ the minimum cost-to-go along the trajectories in $\mathcal{CS}^j$.

8.2.3 Robust LMPC

In this section, we present the proposed robust LMPC algorithm for the linear systems subject to additive uncertainty (8.1). We introduce the following finite time optimal control

problem

$$J^{\text{MPC},j}_{t\to t+N}(x^j_t) = \min_{\bar{x}^j_{t|t},\bar{u}^j_{t|t},\dots,\bar{u}^j_{t+N-1|t}} \quad \sum_{k=t}^{t+N-1} h(\bar{x}^j_{k|t},\bar{u}^j_{k|t}) + Q^j(\bar{x}^j_{t+N|t}) \tag{8.14a}$$

$$\text{s.t.} \quad \bar{x}^j_{k+1|t} = A\bar{x}^j_{k|t} + B\bar{u}^j_{k|t}, \forall k \in \{t,\dots,t+N-1\} \tag{8.14b}$$

$$\bar{x}^j_{k|t} \in \bar{\mathcal{X}},\ \bar{u}^j_{k|t} \in \bar{\mathcal{U}}, \forall k \in \{t,\dots,t+N-1\} \tag{8.14c}$$

$$\bar{x}^j_{t+N|t} \in \mathcal{CS}^j, \tag{8.14d}$$

$$x_t - \bar{x}^j_{t|t} \in \mathcal{E}. \tag{8.14e}$$

The solution to the above optimal control problem is a nominal trajectory which steers the system from $x^{*,j}_{t|t}$ to the $\mathcal{CS}^j$. Problem (8.14) may be used to design a robust MPC, which guarantees robust constraint satisfaction [82]. However, we notice that the first state of the open-loop sequence $x^{*,j}_{t|t}$ is an optimization variable, which enforces $e_t = x_t - \bar{x}^j_{t|t} \in \mathcal{E}$. Therefore, robust MPC strategies based on Problem (8.14) do not guarantee that the certainty equivalent closed-loop trajectory is feasible, i.e. it is not required that $\bar{\mathbf{x}}^j \in \bar{\mathcal{X}}$. For this reason, we introduce the following finite time optimal control problem, which will allow us to guarantee feasibility of the certainty equivalent closed-loop trajectory,

$$J^{\text{LMPC},j}_{t\to t+N}(x^j_t,\bar{x}^j_{t-1},\bar{u}^j_{t-1|t-1}) = \min_{\substack{\bar{u}^j_{t|t},\dots,\bar{u}^j_{t+N-1|t}\\ \bar{x}^j_{t|t},\bar{u}^j_{t-1|t}}} \sum_{k=t}^{t+N-1} h(\bar{x}^j_{k|t},\bar{u}^j_{k|t}) + Q^j(\bar{x}^j_{t+N|t}) \tag{8.15a}$$

$$\text{s.t.} \quad \bar{x}^j_{k+1|t} = A\bar{x}^j_{k|t} + B\bar{u}^j_{k|t}, \forall k \in \{t,\dots,t+N-1\} \tag{8.15b}$$

$$\bar{x}_{k|t} \in \bar{\mathcal{X}},\ \bar{u}^j_{k|t} \in \bar{\mathcal{U}}, \forall k \in \{t,\dots,t+N-1\} \tag{8.15c}$$

$$\bar{x}^j_{t+N|t} \in \mathcal{CS}^j, \tag{8.15d}$$

$$x_t - \bar{x}^j_{t|t} \in \mathcal{E} \tag{8.15e}$$

$$\bar{x}^j_{t|t} = A\bar{x}^j_{t-1} + B\bar{u}^j_{t-1|t} \tag{8.15f}$$

$$h(\bar{x}^j_{t-1},\bar{u}^j_{t-1|t}) \le h(\bar{x}^j_{t-1},\bar{u}^j_{t-1|t-1}). \tag{8.15g}$$

Compare (8.14) with (8.15), we notice that Problem (8.14) has the additional constraints (8.15f) and (8.15g). Those constraints allow us to construct a feasible closed-loop trajectory of the certainty equivalent system which can be used enlarge the safe set. In particular, we define Algorithm 2 which at time $t = 0$ solves Problem (8.14) and for $t \ge 0$ solves Problem (8.15). Then, at each time t of iteration j, we run Algorithm 2 and we apply to system (8.1)

$$u^j_t = -K(x_t - \bar{x}^{j,*}_{t|t}) + \bar{u}^{j,*}_{t|t}. \tag{8.16}$$

Furthermore, we use the output from Algorithm 2 to construct the closed-loop trajectory of the certainty equivalent system

$$\bar{x}^j_t = \bar{x}^{j,*}_{t|t} \tag{8.17}$$

and if $t \geq 0$ we store the associated input action

$$\bar{u}^j_{t-1} = \bar{u}^{j,*}_{t-1|t}. \tag{8.18}$$

The stored input-state pairs from (8.17) and (8.18) represent a feasible trajectory of the certainty equivalent system, which is used to enlarge the safe set and to update the Q-function.

Algorithm 2: LMPC Algorithm

Given the state measured state x^j_t at time t
Given the optimal state $\bar{x}^{j,*}_{t|t-1}$ and input $\bar{u}^{j,*}_{t-1|t-1}$ (Needed if $t > 0$)
if *$t = 1$* **then**
  Let $\bar{x}^{*,j}_{t|t}, \bar{u}^{*,j}_{t|t}, \ldots, \bar{u}^{*,j}_{t+N-1|t}$ be the optimizer to $J^j_{t\to\infty}(\bar{x}^j_t)$
  Return $\bar{x}^{*,j}_{t|t}, \bar{u}^{*,j}_{t|t}$
else
  Let $\bar{x}^{*,j}_{t|t}, \bar{u}^{*,j}_{t-1|t}, \bar{u}^{*,j}_{t|t}, \ldots, \bar{u}^{*,j}_{t+N-1|t}$ be the optimizer to $J^j_{t\to\infty}(\bar{x}^j_t, \bar{x}^{j,*}_{t|t-1}, \bar{u}^{j,*}_{t-1|t-1})$
  Return $\bar{x}^{j,*}_{t|t}$, $\bar{u}^{j,*}_{t|t}$, $\bar{u}^{j,*}_{t-1|t}$
end

8.3 Properties

In this section, we show that the robust LMPC policy (8.16) in closed-loop with system (8.1) guarantees recursive robust constraint satisfaction, asymptotic convergence to a neighborhood of the origin and iterative performance improvement.

Assumption 16 *At iteration 0, we are given a feasible closed-loop trajectory $\bar{\mathbf{x}}^0$ and the associated feasible input sequence $\bar{\mathbf{x}}^0$.*

Theorem 20 *Consider system* (8.1) *in closed loop with the robust LMPC policy* (8.16)*. Let Assumptions 15-16 hold. Then, the closed-loop system* (8.1) *and* (8.16) *satisfies state and input constants and it converges asymptotically to the set $\mathcal{E}$ for every iteration $j \geq 1$ and all $w \in \mathcal{W}$.*

Proof *The proof follows from invariance of $\mathcal{E}$ and the proof of Theorem 4.*

Theorem 21 *Consider system* (8.1) *in closed loop with the robust LMPC policy* (8.16)*. Let Assumptions 15-16 hold. Then, the closed-loop performance of the certainty equivalent system $J^j_{t\to\infty}(\bar{x}^j_S)$ is non-decreasing with the iteration index.*

Proof *The proof follows as in Theorem 5.*

8.4 Example

In this section, we apply the proposed robust LMPC algorithm to robustly steer the system

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t + w_t$$

to a neighborhood of the origin while satisfying state $x_t \in \mathcal{X} = \{x \in \mathbb{R}^2 : ||x||_\infty \leq 15\}$ and input constraint $u_t \in \mathcal{U} = \{u \in \mathbb{R}^1 : ||u||_\infty \leq 5\}$, for all $w_t \in \mathcal{W} = \{w \in \mathbb{R}^2 : ||w||_\infty \leq 0.1\}$. Furthermore, we introduce the following infinite time optimal control problem which we will use to benchmark the performance of the robust LMPC policy,

$$\begin{aligned} J^*_{0\to\infty}(x_S) = \min_{\bar{x}_0, \bar{u}_0, \bar{u}_1, \ldots} \quad & \sum_{t=0}^{\infty} \bar{x}_t^T Q \bar{x}_t + \bar{u}_t^T R \bar{u}_t \\ \text{s.t.} \quad & \bar{x}_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \bar{x}_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t, \forall t \in \{0, 1, \ldots\} \\ & \bar{x}_k \in \bar{\mathcal{X}}, v_k \in \bar{\mathcal{U}}, \forall k \in \{0, 1, \ldots\} \\ & x_S - x_0 \in \mathcal{E}, \end{aligned} \tag{8.19}$$

where $\mathcal{E}$ is a robust positive invariant set for the error dynamics from (8.3) defined by the gain K, which is given by the solution of the LQR problem for $Q = I$ and $R = 10$. Furthermore, the constraint sets $\bar{\mathcal{X}}$ and $\bar{\mathcal{U}}$ are defined as in (8.6).

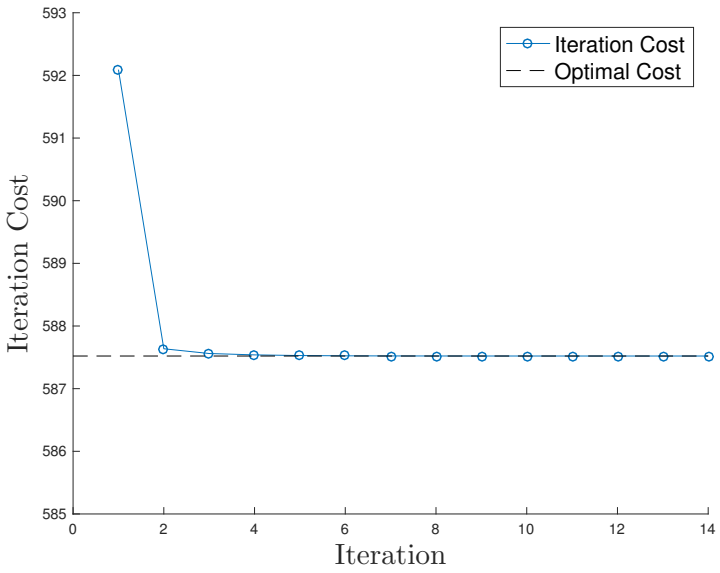


Figure 8.1: Iteration cost at each iteration. We notice that the cost is decreasing until it converges to the optimal cost to Problem (8.19).

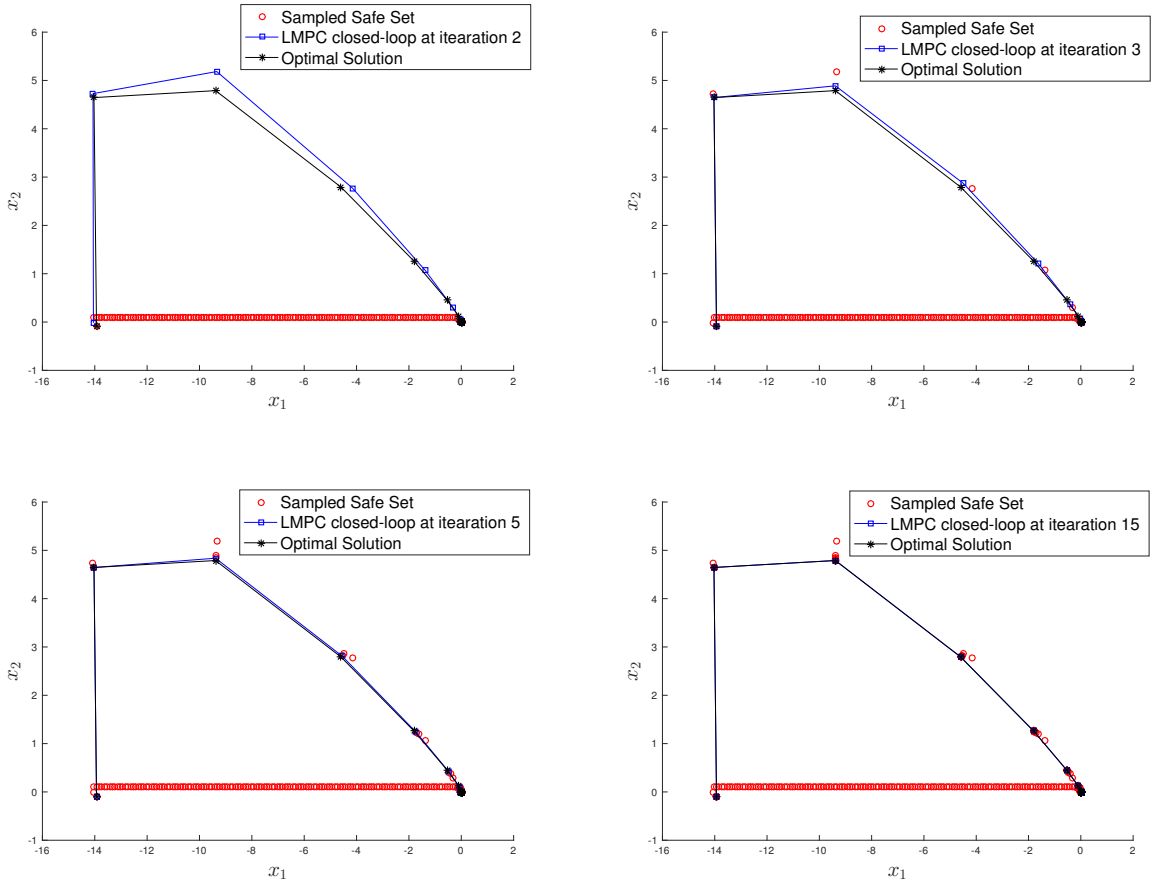


Figure 8.2: Comparison between the optimal trajectory (in black) and the closed-loop trajectories (in blue) at iteration $j = \{2, 3, 5, 15\}$. We notice that as more iterations are performed the sampled safe set (in red) is enlarged and the closed-loop trajectory is closer to the optimal one.

We implement the robust LMPC policy (8.16) with a control horizon $N = 3$. We compute a first feasible solution to initialized the LMPC algorithm and then we use the closed-loop trajectory of the certainty equivalent system to enlarge the safe set and compute the Q-function. Figure 8.2 shows the evolution safe set and closed-loop trajectory for the certainty equivalent system through 4 different iterations. At the first iteration in the top left corner, the controller deviates from the safe set, but the closed-loop trajectory is far from the optimal solution to Problem 8.19. However, as more iteration are performed, the closed-loop trajectories gets closer to the optimal one, until the two trajectories overlap. As a results, the closed-loop cost converges to the optimal one, as shown in Figure 8.1. We notice that the certainty equivalent cost is non-increasing as discussed in Theorem 20.

Finally, we performed 1000 Monte-Carlo simulations for the closed-loop system (8.16) and (8.1), which are shown in Figure 8.3. We confirm that regardless of the disturbance

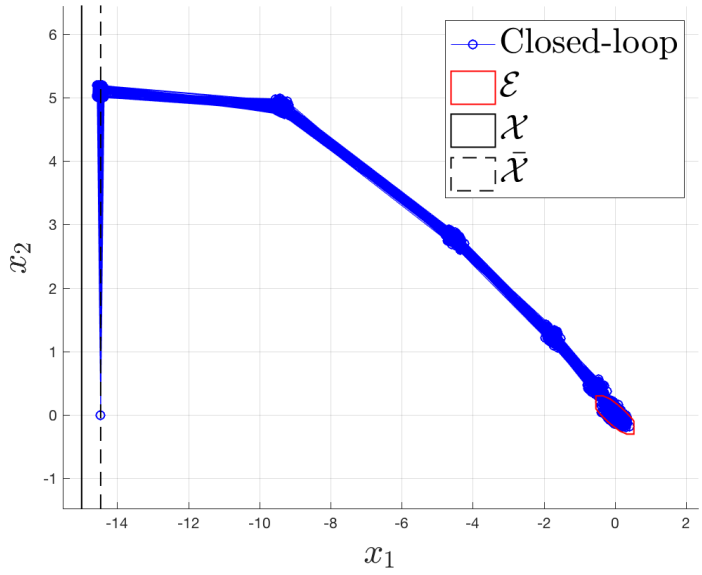


Figure 8.3: Closed-loop simulations for 1000 Monte-Carlo simulations. We notice that the state constraints are robustly satisfied.

realization the robust LMPC policy steered the system to the neighborhood of the origin $\mathcal{E}$, while satisfying state and input constraints.

Chapter 9

Autonomous Racing Experiments

In this chapter, we illustrate the system identification strategy used to implement the Learning Model Predictive Controller (LMPC) for autonomous racing. We model the autonomous racing problem as a minimum time iterative control task, where an iteration corresponds to a lap. The system trajectory and input sequence of each lap are stored and used to systematically update the controller for the next lap. The first part of this chapter, we introduce a local LMPC which reduces the computational burden associated with the strategy proposed in Chapter 4. Afterwards, we present a system identification strategy for the autonomous racing iterative control task. We use data from previous iterations and the vehicle's kinematic equations of motion to build an affine time-varying prediction model. The effectiveness of the proposed strategy is demonstrated by experimental results on the Berkeley Autonomous Race Car (BARC) platform.

9.1 Problem Formulation

Consider the following state and input vectors

$$x = \left[v_x, v_y, w_z, e_\psi, s, e_y\right]^\top \text{ and } u = \left[\delta, a\right]^\top,$$

where w_z, v_x, v_y, are the vehicle's yaw rate, longitudinal and lateral velocities. The position of the vehicle is represented in the curvilinear reference frame [83], where s is the distance travelled along the centerline of the track. The states e_ψ and e_y are the heading angle and lateral distance error between the vehicle and the centerline of the track, as shown in Figure 9.1. Finally, δ and a are the steering and acceleration commands. The vehicle is described by the dynamic bicycle model

$$x_{t+1} = f(x_t, u_t), \tag{9.1}$$

where $f(\cdot, \cdot)$ is derived from kinematics and balancing the forces acting on the tires [67]. A detailed expression can be found in [67, Chapter 2]. Note that in the curvilinear reference

frame state and input constraints are convex, i.e.

$$x_t \in \mathcal{X} = \{x \in \mathbb{R}^n : F_x x \leq b_x\},$$
$$u_t \in \mathcal{U} = \{u \in \mathbb{R}^d : F_u u \leq b_u\},\ \forall t \geq 0.$$

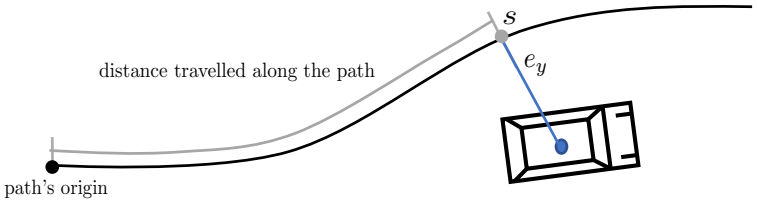


Figure 9.1: Representation of the vehicle's position in the curvilinear reference frame.

The goal of the controller is to drive the system from the starting point x_S to the terminal set $\mathcal{X}_F$. More formally, the controller aims to solve the following minimum time optimal control problem

$$\begin{aligned} \min_{T,u_0,\ldots,u_{T-1}} \quad & \sum_{t=0}^{T-1} 1 \\ \text{s.t.} \quad & x_{t+1} = f(x_t, u_t), && \forall t = [0,\ldots,T-1] \\ & x_t \in \mathcal{X},\ u_t \in \mathcal{U}, && \forall t = [0,\ldots,T] \\ & x_T = \mathcal{X}_F,\ x_0 = x_S, \end{aligned} \tag{9.2}$$

where for a track of length L the terminal set

$$\mathcal{X}_F = \{x \in \mathbb{R}^n : [0\ 0\ 0\ 0\ 1\ 0]x = s \geq L\} \tag{9.3}$$

represents the states beyond the finish line.

9.2 Controller Design

In this section, we first show how to use historical data to construct a terminal constraint set and terminal cost function. Afterwards, we exploit these quantities to design the controller.

9.2.1 Stored Data

We define one iteration as a successful lap around the race track and we store the closed-loop trajectories. In particular, at the jth iteration we define the vectors

$$\begin{aligned} \mathbf{u}^j &= [u_0^j, \ldots,\ u_{T^j}^j] \\ \mathbf{x}^j &= [x_0^j, \ldots,\ x_{T^j}^j], \end{aligned} \tag{9.4}$$

which collect the evolution of closed-loop system and associated input sequence. In the above definitions, T^j denotes the time at which the closed-loop system reached the terminal set, i.e. $x_{T^j} \in \mathcal{X}_F$.

9.2.2 Local Convex Safe Set

This sections shows how to construct a local convex safe set using a subset of the stored data point. In particular, we define the local convex safe set around x is defined as the convex hull of the K-nearest neighbors to x.

First, for the jth trajectory we define the set of time indices $[t_1^{j,*}, \dots, t_K^{j,*}]$ associated with the K-nearest neighbors to the point x,

$$\begin{aligned} [t_1^{j,*}, \dots, t_K^{j,*}] = \underset{t_1,\dots,t_K}{\operatorname{argmin}} \quad & \sum_{i=1}^{K} ||x_{t_i}^j - x||_D^2 \\ \text{s.t.} \quad & t_i \neq t_k, \ \forall i \neq k \\ & t_i \in \{0, \dots, T^j\}, \forall i \in \{1, \dots, K\}. \end{aligned} \tag{9.5}$$

In the above definition $||y||_D^2 = y^\top D^\top D y$ for the user-defined matrix D, which may be chosen to take into account the relative scaling or relevance of different variables. We chose $D = diag(0, 0, 0, 0, 1, 0)$ to select the K-nearest neighbors with respect to the curvilinear abscissa s, which represents a proxy for the distance between two stored data points of the same lap. Furthermore, as the vehicle moves forward on the track, at each lap the stored data are ordered with respect to the travelled distance s and the computation of (9.5) is simplified. The K-nearest neighbors to x from the lth to the jth iteration are collected in the following matrix

$$D_l^j(x) = [x_{t_1^{l,*}}^l, \dots, x_{t_K^{l,*}}^l, \dots, x_{t_1^{j,*}}^j, \dots, x_{t_K^{j,*}}^j],$$

which is used to define the local convex safe set around x

$$\mathcal{CL}_l^j(x) = \{\bar{x} \in \mathbb{R}^n : \exists \boldsymbol{\lambda} \in \mathbb{R}^{K(j-l+1)}, \boldsymbol{\lambda} \geq 0, \mathbb{1}\boldsymbol{\lambda} = 1, D_l^j(x)\boldsymbol{\lambda} = \bar{x}\}. \tag{9.6}$$

Notice that the above local convex safe set $\mathcal{CL}_l^j(x)$ represents the convex hull of the K-nearest neighbors to x from the lth to jth iteration.

Finally, we define the matrix

$$S_l^j(x) = [x_{t_1^{l,*}+1}^l, \dots, x_{t_K^{l,*}+1}^l, \dots, x_{t_1^{j,*}+1}^j, \dots, x_{t_K^{j,*}+1}^j]$$

which collects the evolution of the states stored in the columns of the matrix $D_l^j(x)$. The above matrix $S_l^j(x)$ will be used in Section 9.2.4 to construct the local convex safe set at each time step.

9.2.3 Local Convex Q-function

This section shows how to construct an approximation of the value function over the local convex safe set $\mathcal{CL}_l^j(x)$ around x. In particular, we define the local convex Q-function around x as the convex combination of the cost associated with the stored trajectories,

$$\begin{aligned} Q_l^j(\bar{x}, x) = \min_{\boldsymbol{\lambda}} \quad & \mathbf{J}_l^j(x)\boldsymbol{\lambda} \\ \text{s.t} \quad & \boldsymbol{\lambda} \geq 0, \ \mathbb{1}\boldsymbol{\lambda} = 1, D_l^j(x)\boldsymbol{\lambda} = \bar{x}, \end{aligned} \tag{9.7}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{k(j-l)}$, $\mathbb{1}$ is a row vector of ones and the row vector

$$\mathbf{J}_l^j(x) = [J^l_{t_1^{l,*} \to T^l}(x^l_{t_1^{l,*}}), \dots, J^l_{t_M^{l,*} \to T^l}(x^l_{t_M^{l,*}}), \dots, J^j_{t_1^{j,*} \to T^j}(x^j_{t_1^{j,*}}), \dots, J^j_{t_M^{j,*} \to T^j}(x^j_{t_M^{j,*}})],$$

collects the cost-to-go associated with the K-nearest neighbors to x from the lth the jth iteration. The cost-to-go $J^j_{t \to T^j}(x^j_t) = T^j - t$ represents the time to drive the vehicle from x^j_t to the finish line along the jth trajectory. We underline that the cost-to-go is computed after completion of the jth iteration.

9.2.4 Local LMPC Design

The local convex safe set and the local convex Q-function are used to design the controller. At each time t of the jth iteration the controller solves the following finite time optimal control problem

$$\begin{aligned}
J^{\text{LMPC},j}_{t \to t+N}(x^j_t, z^j_t) = \min_{\mathbf{U}^j_t, \boldsymbol{\lambda}^j_t} \quad & \left[\sum_{k=t}^{t+N-1} h(x^j_{k|t}) + \mathbf{J}_l^{j-1}(z^j_t)\boldsymbol{\lambda}^j_t \right] && \text{(9.8a)} \\
\text{s.t.} \quad & x^j_{t|t} = x^j_t, && \text{(9.8b)} \\
& \boldsymbol{\lambda}^j_t \geq 0, \mathbb{1}\boldsymbol{\lambda}^j_t = 1, D_l^{j-1}(z^j_t)\boldsymbol{\lambda}^j_t = x^j_{t+N|t} && \text{(9.8c)} \\
& x^j_{k+1|t} = A^j_{k|t}x^j_{k|t} + B^j_{k|t}u^j_{k|t} + C^j_{k|t}, && \text{(9.8d)} \\
& x^j_{k|t} \in \mathcal{X}, u^j_{k|t} \in \mathcal{U}, && \text{(9.8e)} \\
& \forall k = t, \cdots, t+N-1,
\end{aligned}$$

where $\mathbf{U}^j_t = [u^j_{t|t}, \dots, u^j_{t+N-1|t}] \in \mathbb{R}^{d \times N}$, $\boldsymbol{\lambda}^j_t \in \mathbb{R}^{(j-l+1)K}$ and the stage cost in (9.8a)

$$h(x) = \begin{cases} 1 & \text{If } x \notin \mathcal{X}_F \\ 0 & \text{Else} \end{cases}.$$

In the above finite time optimal control problem equations (9.8b), (9.8d) and (9.8e) represent the dynamic update, state and input constraints. Finally, (9.8c) enforces $x^j_{t+N|t}$ into the local convex safe set defined in Section 9.2.2. The optimal solution to (9.8) at time t of the jth iteration

$$\boldsymbol{\lambda}^{j,*}_t, [x^{j,*}_{t|t}, \dots, x^{j,*}_{t+N|t}] \text{ and } \mathbf{U}^{j,*}_t = [u^{j,*}_{t|t}, \dots, u^{j,*}_{t+N-1|t}] \tag{9.9}$$

is used to compute the following vector

$$z^j_t = \begin{cases} x^{j-1}_N & \text{If } t = 0 \\ S^j_l(z^j_{t-1})\boldsymbol{\lambda}^{j,*}_{t-1} & \text{Otherwise} \end{cases}, \tag{9.10}$$

which at time t defines the local convex safe set $\mathcal{LS}_l^j(z_t^j)$ and local Q-function $Q_l^j(x, z_t^j)$ in (9.8). The above vector z_t^j represents a candidate terminal state for the planned trajectory of the LMPC at time t. First, we initialize the candidate terminal state z_0^j using the $(j-1)$th trajectory. Afterwards, we update the vector z_t^j as the convex combination of the columns of the matrix $S_l^j(z_t^j)$ from Section 9.2.2. Notice that if the systems is linear or if a linearized system approximates the nonlinear dynamics over the local convex safe set, then there exists a feasible input which drives the system from $x_{t+N|t}^{j,*} = D_t^{j-1}(z_t^j)\boldsymbol{\lambda}_t^{j,*}$ to $z_{t+1}^j = S_l^{j-1}(z_t^j)\boldsymbol{\lambda}_t^{j,*}$.

Finally, we apply to the system (9.1) the first element of the optimizer vector,

$$u_t^j = u_{t|t}^{j,*}. \tag{9.11}$$

The finite time optimal control problem (9.8) is repeated at time $t+1$, based on the new state $x_{t+1|t+1} = x_{t+1}^j$.

9.3 System Identification Strategy

We illustrate the system identification strategy used to build an Affine Time Varying (ATV) model which approximates the vehicle dynamics. First, we introduce the kinematic equations of motion which describe the evolution of the vehicle's position as a function of the velocities. Afterwards, we present the strategy used to approximate the dynamic equations of motion, which model the evolution of the vehicle's velocities as a function of the input commands. Finally, we describe the ATV model, which is computed online linearizing the kinematic equations of motion and evaluating the approximate dynamic equations of motion along the shifted optimal solution to the LMPC.

9.3.1 Kinematic Model

As mentioned in Section 9.1, the position of the vehicle is expressed in the Frenet reference frame [83]. In particular, we describe the position of the vehicle in terms of lateral distance e_y from the centerline of the road and distance s traveled along a predefined path (Fig. 9.1). The state e_ψ represents the difference between the vehicle's heading angle and the angle of the tangent vector to the path at the curvilinear abscissa s.

The rate of change of the vehicle's position in the curvilinar reference frame is described by the following kinematic relationships

$$\begin{aligned}
\dot{e}_\psi &= w_z - \frac{v_x \cos(e_\psi) - v_y \sin(e_\psi)}{1 - \kappa(s) e_y} \kappa(s) \\
\dot{s} &= \frac{v_x \cos(e_\psi) - v_y \sin(e_\psi)}{1 - \kappa(s) e_y} \\
\dot{e}_y &= v_x \sin(e_\psi) + v_y \cos(e_\psi),
\end{aligned}$$

where $\kappa(s)$ is the curvature of the centerline of the track at the curvilinear abscissa s [83]. The above equations can be Euler discretized to approximate the vehicle's motion as a function of the vehicle's velocities

$$\begin{aligned} e_{\psi_{k+1}} &= f_{e_\psi}(x_k) = e_{\psi_k} + dt\left(w_{z_k} - \frac{v_{x_k}\cos(e_{\psi_k}) - v_{y_k}\sin(e_{\psi_k})}{1-\kappa(s_k)e_{y_k}}\kappa(s_k)\right) \\ s_{k+1} &= f_s(x_k) = s_k + dt\left(\frac{v_{x_k}\cos(e_{\psi_k}) - v_{y_k}\sin(e_{\psi_k})}{1-\kappa(s_k)e_{y_k}}\right) \\ \dot{e}_y &= f_{e_y}(x_k) = e_{y_k} + dt\left(v_{x_k}\sin(e_{\psi_k}) + v_{y_k}\cos(e_{\psi_k})\right), \end{aligned} \tag{9.12}$$

where dt is the discretization time. The above equations will be linearized to compute an ATV prediction model. It is interesting to notice that equations (9.12) are independent of the vehicle's physical parameters, because these are derived from kinematic relationships between velocities and position.

9.3.2 Dynamic Model

The dynamic equations of motion, which describe the evolution of the vehicle's velocities, may be computed balancing the forces acting on the tires [67]. Therefore, the dynamic equations depend on physical parameters associated with the vehicle, tires and asphalt. These parameters may be estimated through a system identification campaign. However, the nonlinear dynamic equations of motion should be linearized in order to obtain an ATV model which allows us to reformulate the LMPC as a QP. Instead of identifying the parameters of a nonlinear model and then linearize it, we propose to directly learn a linear model around x using a local linear regressor. We introduce the Epanechnikov kernel function [84]

$$K(u) = \begin{cases} \frac{3}{4}(1-u^2), & \text{for } |u| < 1 \\ 0, & \text{else} \end{cases},$$

which is used to compute a local linear model around x for the longitudinal and lateral dynamics. In particular, for $l = \{v_x, v_y, w_z\}$ we compute the following regressor vector

$$\Gamma^l(x) = \underset{\Gamma}{\operatorname{argmin}} \sum_{\{k,j\}\in I(x)} K\left(\frac{||x - x_k^j||_Q^2}{h}\right) y_k^{j,l}(\Gamma), \tag{9.13}$$

where the hyperparameter $h \in \mathbb{R}_+$ is the bandwidth, the row vector $\Gamma \in \mathbb{R}^5$,

$$\begin{aligned} y_k^{j,v_x}(\Gamma) &= ||v_{x_{k+1}}^j - \Gamma[v_{x_k}^j,\ v_{y_k}^j,\ w_{z_k}^j,\ a_k^j,\ 1]^T|| \\ y_k^{j,v_y}(\Gamma) &= ||v_{y_{k+1}}^j - \Gamma[v_{x_k}^j,\ v_{y_k}^j,\ w_{z_k}^j,\ \delta_k^j,\ 1]^T|| \\ y_k^{j,w_z}(\Gamma) &= ||w_{z_{k+1}}^j - \Gamma[v_{x_k}^j,\ v_{y_k}^j,\ w_{z_k}^j,\ \delta_k^j,\ 1]^T||, \end{aligned}$$

and $I_l^j(x)$ is the set of indices

$$I_l^j(x) = \underset{\{k_1,j_1\},\ldots,\{k_P,j_P\}}{\operatorname{argmin}} \quad \sum_{i=1}^{P} ||x - x_{k_i}^{j_i}||_Q^2$$
$$\text{s.t.}$$
$$k_i \neq k_n,\ \forall j_i = j_n$$
$$k_i \in \{1, 2, \ldots\}, \forall i \in \{1, \ldots, P\}$$
$$j_i \in \{l, \ldots, j\}, \forall i \in \{1, \ldots, P\},$$

where $||y||_Q = y^\top Q^\top Q y$ and the matrix Q is user defined. For the stored data from iteration l to iteration j, the set $I_l^j(x)$ collects the indices associated with the P-nearest neighbors to the state x. Finally, the user-defined matrix Q takes into account the relative scaling of different variables.

Notice that the optimizer in (9.13) can be used to approximate the evolution of vehicle's velocities,

$$\begin{bmatrix} v_{x_{k+1}} \\ v_{y_{k+1}} \\ w_{z_{k+1}} \end{bmatrix} = \begin{bmatrix} \Gamma_{1:3}^{v_x}(x) \\ \Gamma_{1:3}^{v_y}(x) \\ \Gamma_{1:3}^{w_z}(x) \end{bmatrix} \begin{bmatrix} v_{x_k} \\ v_{y_k} \\ w_{z_k} \end{bmatrix} + \begin{bmatrix} \Gamma_4^{v_x}(x) & 0 \\ 0 & \Gamma_4^{v_y}(x) \\ 0 & \Gamma_4^{w_z}(x) \end{bmatrix} \begin{bmatrix} a_k \\ \delta_k \end{bmatrix} + \begin{bmatrix} \Gamma_5^{v_x}(x) \\ \Gamma_5^{v_y}(x) \\ \Gamma_5^{w_z}(x) \end{bmatrix}, \tag{9.14}$$

where for $l = \{v_x, v_y, w_z\}$ the scalar $\Gamma_i^l(x)$ denotes the ith element of the vector $\Gamma^l(x)$ and $\Gamma_{1:3}^l(x) \in \mathbb{R}^3$ is a row vector collecting the first three elements of $\Gamma^l(x)$ in (9.13).

9.3.3 Affine Time Varying Model

In this section we describe the strategy used to build an ATV model, which is then used for control. At time t of the jth iteration, we define the candidate solution $\bar{\mathbf{x}}_t^j = [\bar{x}_{t|t}^j, \ldots, \bar{x}_{t+N|t}^j]$ to Problem (9.8) using the optimal solution at time $t-1$ from (9.9),

$$\bar{x}_{k|t}^j = \begin{cases} x_{k|t-1}^{j,*} & \text{If } k \in \{t, \ldots, t+N-1\} \\ z_t^j & \text{If } k = t+N \end{cases}.$$

Finally at each time t of iteration j, the above candidate solution is used to build the following ATV model

$$x_{k+1|t}^j = A_{k|t}^j x_{k|t}^j + B_{k|t}^j u_{k|t}^j + C_{k|t}^j, \tag{9.15}$$

where $x^j_{k|t} = [v^j_{x_{k|t}}, v^j_{y_{k|t}}, w^j_{y_{k|t}}, e^j_{\psi_{k|t}}, s^j_{k|t}, e^j_{y_{k|t}}]$ and the matrices $A^j_{k|t}$, $B^j_{k|t}$ and $C^j_{k|t}$ are obtained linearizing (9.12) around $\bar{x}^j_{k|t}$ and evaluating (9.14) at $\bar{x}^j_{k|t}$,

$$A^j_{k|t} = \begin{bmatrix} \Gamma^{v_x}_{1:3}(\bar{x}^j_{k|t}) & 0 & 0 & 0 \\ \Gamma^{v_y}_{1:3}(\bar{x}^j_{k|t}) & 0 & 0 & 0 \\ \Gamma^{w_z}_{1:3}(\bar{x}^j_{k|t}) & 0 & 0 & 0 \\ (\nabla_x f_{e_\psi}(x)|_{\bar{x}^j_{k|t}})^\top \\ (\nabla_x f_s(x)|_{\bar{x}^j_{k|t}})^\top \\ (\nabla_x f_{e_y}(x)|_{\bar{x}^j_{k|t}})^\top \end{bmatrix}, B^j_{k|t} = \begin{bmatrix} \Gamma^{v_x}_4(\bar{x}^j_{k|t}) & 0 \\ 0 & \Gamma^{v_y}_4(\bar{x}^j_{k|t}) \\ 0 & \Gamma^{w_z}_4(\bar{x}^j_{k|t}) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{9.16}$$

and

$$C_k = \begin{bmatrix} \Gamma^{v_x}_5(\bar{x}^j_{k|t}) \\ \Gamma^{v_y}_5(\bar{x}^j_{k|t}) \\ \Gamma^{w_z}_5(\bar{x}^j_{k|t}) \\ f_{e_y}(\bar{x}^j_{k|t}) - (\nabla_x f_{e_y}(x)|_{\bar{x}^j_{k|t}})^\top \bar{x}^j_{k|t} \\ f_s(\bar{x}^j_{k|t}) - (\nabla_x f_s(x)|_{\bar{x}^j_{k|t}})^\top \bar{x}^j_{k|t} \\ f_{e_\psi}(\bar{x}^j_{k|t}) - (\nabla_x f_{e_\psi}(x)|_{\bar{x}^j_{k|t}})^\top \bar{x}^j_{k|t} \end{bmatrix} . \tag{9.17}$$

9.4 Experiments

The proposed control strategy has been implemented on a 1/10-scale open source vehicle platform called Berkeley Autonomous Race Car[1] (BARC). The vehicle is equipped with a set of sensors, actuators and two on-board CPUs to perform low-level control of the actuators as well as communication with a laptop, on which the high-level control is implemented. The on board sensors are magnetically operated encoders, an ultrasound-based indoor positioning system and an Inertial Measurement Unit (IMU). The CPUs are an Arduino Nano for low-level control of the actuators and an Odroid XU4 for WiFi communication with the i7 MSI GT72 laptop. The actuators are an electrical motor and a servo for the steering. The control architecture has been implemented in the Robot Operating System (ROS) framework, using Python and OSQP [85]. The code is available online[2]

We initialize the algorithm performing two laps of path following at constant speed. Each jth iteration collects the data of two consecutive laps. Therefore, the local safe set and local Q-function are defined also beyond the finish line. This strategy allows us to implement the LMPC for the repetitive autonomous racing control task, as shown in [86]. At each jth lap, we use the LMPC (9.8) and (9.11) to drive the vehicle from the starting line to the finish line and we use the closed-loop data to update the controller for the next lap. The parameters

[1] A video of the experiment can be found at `https://youtu.be/ZBFJWtIbtMo`

[2] The code is available on the BARC GitHub repository in the "devel-ugo"

which define the controller are reported in Table 9.1. We also added a small input rate cost in order to guarantee a unique solution to the QP associated with the LMPC.

Table 9.1: Parameters used in the controller design.

l	$j-2$
K	20
T	$\text{diag}(0,0,0,0,1,0)$
Q	$\text{diag}(0.1,1,1,0,0,0)$
P	80
h	10
N	12

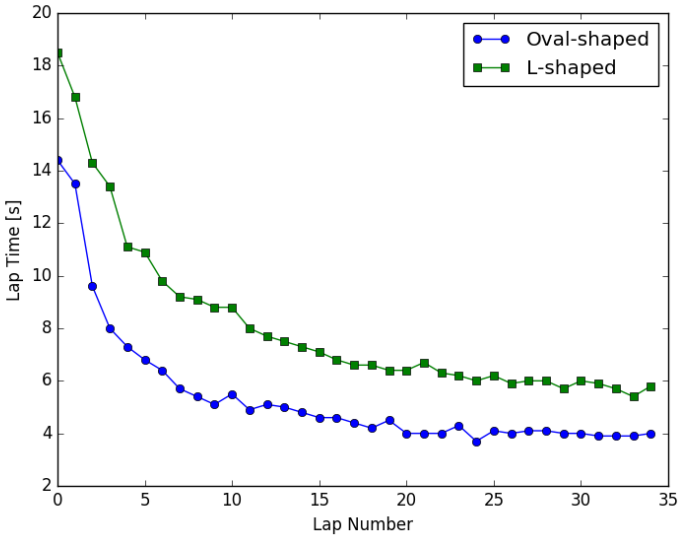


Figure 9.2: Lap time of the LMPC on the oval-shaped and L-shaped tracks.

We tested the controller on an oval-shaped and L-shaped tracks on which the vehicle runs in the counter-clockwise direction. Figure 9.2 shows that the lap time decreases until convergence is reached after 29 laps. Furthermore, Figure 9.3 shows the evolution of the closed-loop trajectory on the X-Y plane and the velocity profile which is color coded. In the first row we reported the path following trajectory used to initialize the LMPC and the closed-loop trajectories at laps 7 and 15. We notice that the controller deviates from the initial feasible trajectory (reported in blue as the vehicle speed is 1.2m/s) in order to explore the state space and to drive the vehicle at higher speeds, until it converges to a steady-state behavior. The steady-state trajectories from lap 30 to 34 are reported in the bottom row of Figure 9.3. Notice that the color bar representing the velocity profile changed from the first

to second row as the vehicle runs at higher speed at the end of the learning process. We underline that the controller understands the benefit of breaking right before entering the curve and of accelerating when exiting. This behavior is optimal in racing as shown in [87].

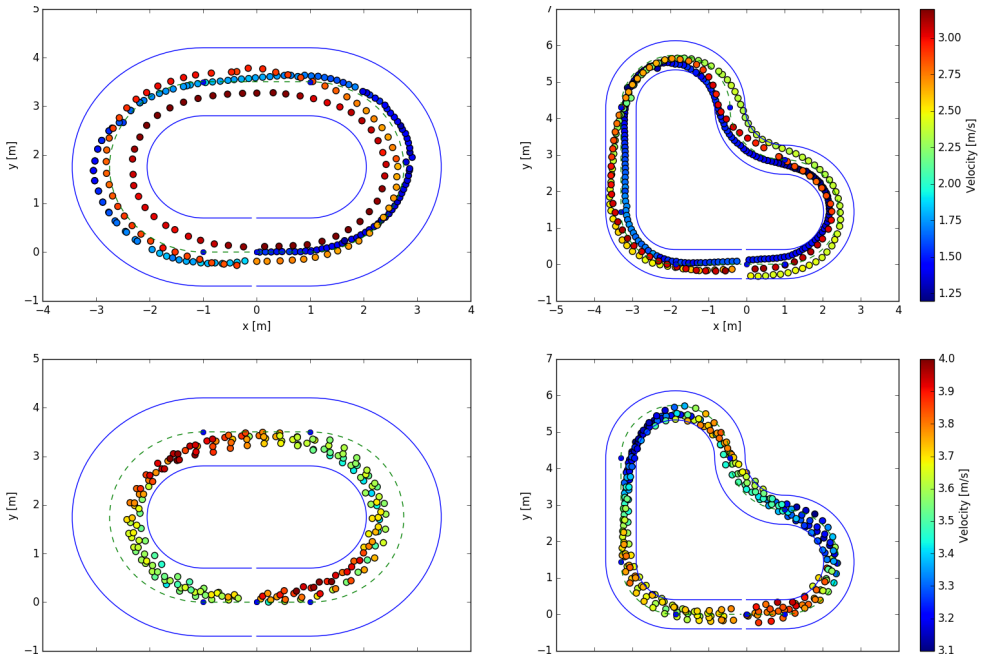


Figure 9.3: The first row in the above figure shows the closed-loop trajectory used to initialize the LMPC and the closed-loop trajectories after few laps of learning. The second row shows the steady state trajectories at which the LMPC has converged. Notice that the scale of the color bar changes from the first to the second row, as the vehicle runs at higher speed after the learning process has converged.

Figure 9.4 shows the raw acceleration measurements from the IMU. We confirm that controller is able to operate the vehicle at the limit of its handling capability, reaching a maximum lateral acceleration close to 1g [3]. Furthermore, Figure 9.5 shows the data points used to design the LMPC. Recall from Table 9.1 that at the jth lap the LMPC policy is synthesized using the trajectories from lap $l = j-2$ to lap $j-1$. Therefore, as the controller drives faster on the track, less data points are needed to design the LMPC. Moreover, in Figure 9.6 we reported the computational time. It is interesting to notice that on average the finite time optimal control problem (9.8) is solved in less then 10ms. Finally, we notice that it would be possible to parallelize the computation of the $N-1$ linear models which define the ATV model from (9.15). Indeed, at time t Equations (9.16)-(9.17) may be evaluated independently and in parallel for each predicted time k.

[3]The maximum allowed lateral acceleration is computed assuming that the aerodynamic effects are negligible and the that lateral force acting on the vehicle is $F = \mu mg$ for the friction coefficient $\mu = 1$.

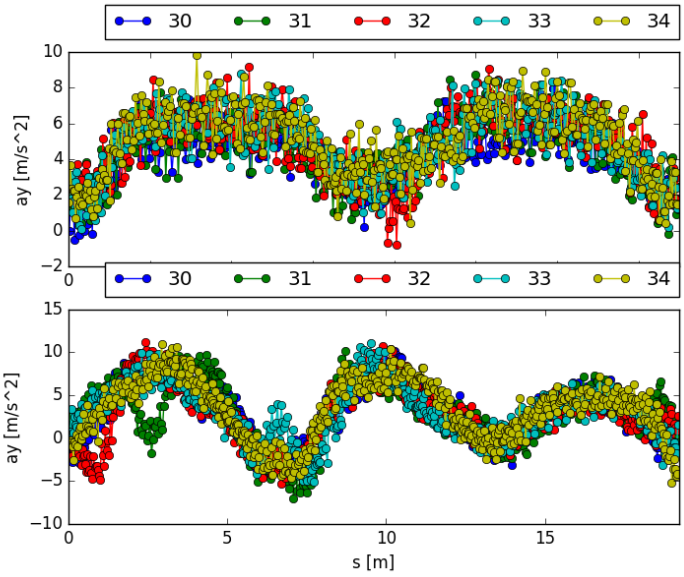


Figure 9.4: Recorded lateral acceleration of the vehicle running on the oval-shaped track (top row) and L-shaped track (bottom row).

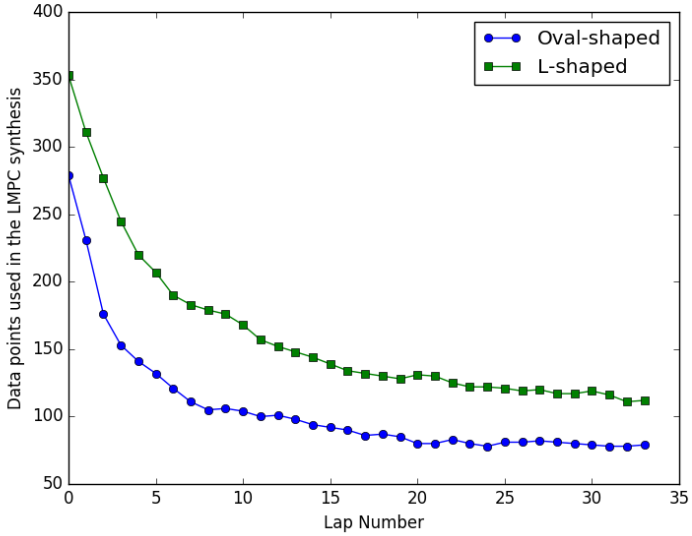


Figure 9.5: Data points used in the LMPC design at each lap.

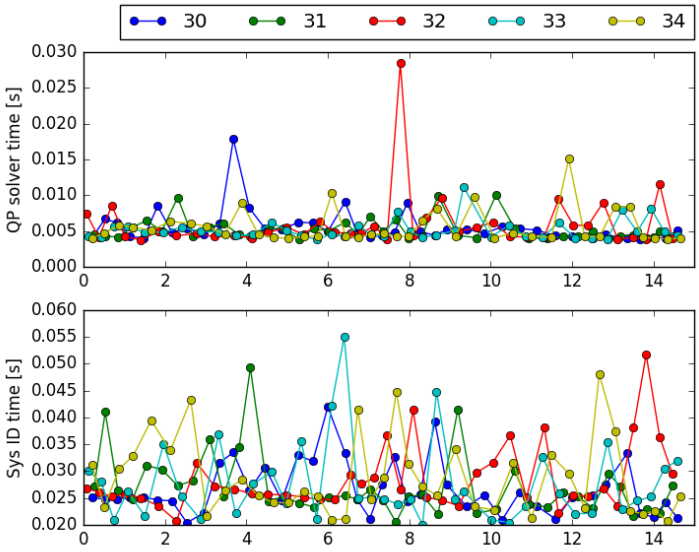


Figure 9.6: The first rows shows the computational cost associated with the FTOCP. In the second row we reported the computational cost associated with the system identification strategy.

Chapter 10

Data-Based Policy

As we have discussed in the previous chapters, the control action given by the LMPC policy is computed solving a finite time optimal control problem over a moving time horizon. This receding horizon strategy allows the controller to deviate from the previous iterations of the control task in order to improve the closed-loop performance. In this chapter, we show how to synthesize a data-based policy, which does not explore the state space, but it is able to math the closed-loop performance of the trajectories used in the synthesis process. Therefore, this strategy may be used to reduce the computational burden of the LMPC once the learning process has converged.

The proposed strategy is model-free and can be applied whenever safe input and state trajectories of a system performing an iterative task are available. These trajectories, together with a user-defined cost function, are exploited to construct a piecewise affine approximation to the value function. The approximated value function is then used to evaluate the control policy by solving a linear program. We show that for linear system subject to convex cost and constraints, the proposed strategy guarantees closed-loop constraint satisfaction and performance bounds for the closed-loop trajectory. We evaluate the proposed strategy in simulations and experiments, the latter carried out on the Berkeley Autonomous Race Car (BARC) platform. We show that the proposed strategy is able to reduce the computation time associated with the LMPC policy by one order of magnitude while achieving the same performance as our model-based control algorithm.

10.1 Problem Formulation

Consider the unknown deterministic system

$$x_{t+1} = Ax_t + Bu_t \tag{10.1}$$

where $x_t \in \mathbb{R}^n$ and $u_t \in \mathbb{R}^d$ are the system's state and input, respectively. Furthermore, the system is subject to the following state and input constraints,

$$x_t \in \mathcal{X} \text{ and } u_t \in \mathcal{U},\ \forall t \in \{0, \dots, T\} \tag{10.2}$$

where T is the time as which the control task is completed.

In the following we assume that closed-loop state and input trajectories starting at different initial states x_0 are stored. In particular, for $j \in \{0, \dots, M\}$ we are given the following input sequences

$$\mathbf{u}^j = [u_0^j, \dots, u_{T^j}^j] \tag{10.3}$$

and the associated closed-loop trajectories

$$\mathbf{x}^j = [x_0^j, \dots, x_{T^j}^j] \tag{10.4}$$

where $x_{t+1}^j = Ax_t^j + Bu_t^j$ and T^j is the time at which the task is completed. These trajectories will be used to design a data-based policy for the unknown system (10.1).

Finally, we defined the cost-to-go associated with the jth closed-loop trajectory

$$J^j\big(x_0^j\big) = \sum_{t=0}^{T^j} h(x_t^j, u_t^j), \tag{10.5}$$

where x_t^j and u_t^j are the stored state and applied input to system (10.1) at time t of the jth iteration.

Assumption 17 *All $M+1$ input and state sequences in* (10.3)-(10.4) *are feasible and known. Furthermore, assume that the state sequence in* (10.4) *converges to the origin and the terminal input $u_{T^j}^j = 0$.*

Remark 11 *We have decided to focus on the linear systems* (10.1) *as this will allow us to rigorously characterize the properties of the proposed approach. However, we underline that the computational cost associated with the proposed strategy is independent on the linearity of the controlled system. Thus, the proposed strategy can be implemented also on nonlinear systems as shown in Section 10.4.2.*

Remark 12 *We have decided to consider a regulation problem to streamline the presentation of the paper. In the Appendix, we show that the proposed strategy can be used to steer system* (10.1) *to a terminal control invariant set $\mathcal{X}_F$, without losing guarantees on safety and performance.*

10.2 Proposed Approach

In this section we describe the proposed approach. First, we recall the definition of the sampled safe set and value function approximation computed from data, which were first introduced in Chapter 4. Afterwards, we show how these quantities are used to evaluate the data-based policy.

10.2.1 Safe Set

We define the collection of the M closed-loop trajectories in (10.4) as the sampled *Safe Set*,

$$\mathcal{SS}^M = \bigcup_{j=0}^{M} \bigcup_{t=0}^{T^j} x_t^j.$$

Notice that for all $x \in \mathcal{SS}^M$, it exists a sequence of control actions that can steer the system to the origin [62]. Finally, we define the *convex safe set* $\mathcal{CS}^M$ as

$$\mathcal{CS}^M = \text{Conv}\big(\mathcal{SS}^M\big). \tag{10.6}$$

$\mathcal{CS}^M$ will be used in the next section to defined the domain of the approximation to the value function.

10.2.2 Q-function

In this section we show how the stored data in (10.3) and (10.4) are used to approximate the value function. First, given the stored states $\mathbf{x}^j$ and inputs $\mathbf{u}^j$ for $j \in \{0, \dots, M\}$, we define the cost-to-go associated with each stored state x_t^j,

$$J_t^j(x_t^j) = \sum_{k=t}^{T^j} h(x_k^j, u_k^j).$$

The realized cost-to-go is used to compute the *Q-function* defined as

$$\begin{aligned} Q^M(x) = \min_{\boldsymbol{\lambda} \geq 0} \quad & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_k^j J_k^j(x_k^j) \\ \text{s.t.} \quad & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_k^j = 1, \\ & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_k^j x_k^j = x. \end{aligned} \tag{10.7}$$

where $\boldsymbol{\lambda} = [\lambda_0^0, \dots, \lambda_{T^0}^0, \dots, \lambda_0^M, , \dots, \lambda_{T^M}^M]$. The Q-function $Q^M(\cdot)$ interpolates the realized cost-to-go over the convex safe set. Moreover, we underline that Problem (10.7) is a parametric LP and therefore $Q^M(x)$ is a piecewise affine function of x [10]. Finally, we notice that the domain of $Q^M(\cdot)$ is the convex safe set $\mathcal{CS}^M$, indeed $\forall x \notin \mathcal{CS}^M$ the optimization problem (10.7) is not feasible.

10.2.3 Data-Based Policy

We are finally ready to introduce the data-based policy. At each time t, we evaluate the approximation to the value function (10.7) at the current state x_t, solving the following optimization problem,

$$\begin{aligned} Q^M(x_t) = \min_{\boldsymbol{\lambda}_t \geq 0} \quad & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_{k|t}^j J_k^j(x_k^j) \\ \text{s.t.} \quad & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_{k|t}^j = 1, \\ & \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_{k|t}^j x_k^j = x_t. \end{aligned} \tag{10.8}$$

where $\boldsymbol{\lambda}_t = [\lambda_{0|t}^0, \dots, \lambda_{T^0|t}^0, \dots, \lambda_{0|t}^M, \dots \lambda_{T^M|t}^M]$.
Let

$$\boldsymbol{\lambda}_t^* = [\lambda_{0|t}^{0,*}, \dots, \lambda_{k|t}^{j,*}, \dots, \lambda_{T^M|t}^{M,*}] \tag{10.9}$$

be the optimal solution at time t to (10.8), then we apply to system (10.1) the following input

$$u_t = \pi(x_t) = \sum_{j=0}^{M} \sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} u_k^j. \tag{10.10}$$

Basically, the data-based policy (10.8) and (10.10) computes the control input u_t as the weighted sum of stored inputs, where the weights are the solution to the minimization problem (10.8).

10.2.4 Local Data-Based Policy

In this section, we propose a Local Data-Based policy which can be used to limit the computational burden of problem (10.8), when a considerable amount of data is given. First, we define the local Q-function $Q_L^M(\cdot)$ as

$$\begin{aligned} Q_L^M(x_t) = \min_{\boldsymbol{\lambda}_t \geq 0} \quad & \sum_{j=0}^{M} \sum_{k \in \mathcal{K}^j(x)} \lambda_{k|t}^j J_k^j(x_k^j) \\ \text{s.t.} \quad & \sum_{j=0}^{M} \sum_{k \in \mathcal{K}^j(x)} \lambda_{k|t}^j = 1, \\ & \sum_{j=0}^{M} \sum_{k \in \mathcal{K}^j(x)} \lambda_{k|t}^j x_k^j = x_t \end{aligned} \tag{10.11}$$

where $\boldsymbol{\lambda}_t = [\lambda^0_{t_1^{0,*}|t}, \dots, \lambda^0_{t_N^{0,*}|t}, \dots, \lambda^0_{t_1^{M,*}|t}, \dots, \lambda_{t_N^{M,*}|t}]$. The elements of the set $\mathcal{K}^j(x) = \{t_1^{j,*}, \dots, t_K^{j,*}\}$ are defined as

$$\begin{aligned}[t_1^{j,*}, \dots, t_K^{j,*}] = \underset{t_1,\dots,t_K}{\operatorname{argmin}} \quad & \sum_{i=1}^{K} ||x^j_{t_i} - x||^2_D \\ \text{s.t.} \quad & t_i \neq t_k, \ \forall i \neq k \\ & t_i \in \{0, \dots, T^j\}, \forall i \in \{1, \dots, K\}.\end{aligned}$$

For the j-th trajectory, the set $\mathcal{K}^j(x)$ collects the indices of the N closest point to the state x. Notice that $K \leq \max_{i\in\{0,\dots,j\}} T^i$ is a user-defined parameter.

Finally, we define the local data-based policy where at each time t we solve $Q^M_L(x_t)$ in (10.11). Then, given the optimal solution $\boldsymbol{\lambda}^*_t$ to Problem (10.11), we apply the following input

$$u_t = \pi(x_t) = \sum_{j=0}^{M} \sum_{k\in\mathcal{K}^j(x_t)} \lambda^{j,*}_{k|t} u^j_k \tag{10.12}$$

to system (10.1).

10.3 Properties

In this section we analyze the properties of the proposed data-based policy (10.8) and (10.10). We show that the proposed strategy guarantees safety, closed-loop stability and performance bounds.

Proposition 11 *(Feasibility) Consider the closed-loop system* (10.1) *and* (10.10)*. Let Assumptions 17 hold and* $\mathcal{CS}^M$ *be the convex safe set defined in* (10.6)*. If the initial state* $x_0 \in \mathcal{CS}^M$*. Then, the data-based policy* (10.8) *and* (10.10) *is feasible for all time* $t \geq 0$*.*

Proof *The proof follows from linearity of the system.*
We assume that at time t *the system state* $x_t \in \mathcal{CS}^M$*, therefore the optimization problem* (10.8) *is feasible. Let* (10.9) *be the optimal solution to* (10.8)*, then at the next time step* $t+1$ *we have*

$$\begin{aligned} x_{t+1} &= Ax_t + B\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda^{j,*}_{k|t} u^j_k \\ &= A\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda^{j,*}_{k|t} x^j_k + B\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda^{j,*}_{k|t} u^j_k \\ &= \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda^{j,*}_{k|t}(Ax^j_k + Bu^j_k) \in \mathcal{CS}^M. \end{aligned}$$

By Assumption 17 we have that

$$\sum_{j=0}^{M} \lambda^{j,*}_{T^j|t}(Ax^j_{T^j} + Bu^j_{T^j}) = 0$$

and therefore

$$x_{t+1} = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda^{j,*}_{k|t}(Ax^j_k + Bu^j_k) = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \bar{\lambda}^j_k x^j_k$$

where $\forall j \in \{0, \dots M\}$

$$\begin{aligned} &\bar{\lambda}^j_0 = 0, \\ &\bar{\lambda}^j_{k_j} = \lambda^{j,*}_{k_j-1|t}, \qquad\qquad \forall k_j \in \{1, \dots, T^j - 1\} \\ &\bar{\lambda}^j_{T^j} = \lambda^{j,*}_{T^j-1|t} + \lambda^{j,*}_{T^j|t} \end{aligned} \tag{10.13}$$

is a feasible solution to the optimization problem (10.8) *at time* $t+1$.
By assumption we have that at time $t = 0$ *the state* $x_0 \in \mathcal{CS}^M$. *Furthermore, we have shown that if at time* t *the state* $x_t \in \mathcal{CS}^M$, *then at time* $t+1$ *the state* $x_{t+1} \in \mathcal{CS}^M$ *and the optimization problem* (10.8) *is feasible. Therefore by induction we conclude that* $x_t \in \mathcal{CS}^M \subseteq \mathcal{X},\ \forall t \in \mathbb{Z}_{0+}$ *and that the optimization problem* (10.8) *is feasible* $\forall t \in \mathbb{Z}_{0+}$.

The above *Proposition 1* implies that the data-based policy (10.8) and (10.10) satisfies the input constraints, and the closed-loop system (10.1) and (10.10) satisfies the state constraints at all time instants, i.e. $u_t \in \mathcal{U}$ and $x_t \in \mathcal{X},\ \forall t \in \mathbb{Z}_{0+}$.

Assumption 18 *The stage cost* $h(\cdot,\cdot)$ *is a continuous convex function and* $\forall u \in \mathcal{U}$ *it satisfies*

$$h(0,u) = 0, \text{ and } h(x,u) \succ 0 \ \forall\ x \in \mathbb{R}^n \setminus \{0\}.$$

Proposition 12 *(Convergence) Consider the closed-loop system* (10.1) *and* (10.10). *Let Assumptions 17-18 hold and* $\mathcal{CS}^M$ *be the convex safe set defined in* (10.6). *If the initial state* $x_0 \in \mathcal{CS}^M$. *Then, the origin of the closed-loop system* (10.1) *and* (10.10) *is asymptotically stable.*

Proof *In the following we show that the approximated value function* $Q^M(\cdot)$ *from* (10.8) *is a Lyapunov function for the origin of the closed loop system* (10.1) *and* (10.10). *Continuity of* $Q^M(\cdot)$ *can be shown as in [10, Chapter 7]. Moreover from* (10.5) *and Assumption 2 we have that* $Q^M(x) \succ 0\ \forall\ x \in \mathcal{CS}^M \setminus \{0\}$ *and* $Q^M(0) = 0$. *Thus, we need to show that* $Q^M(\cdot)$ *is decreasing along the closed loop trajectory.*

By feasibility of Problem (10.8) *from Theorem 1, we have that at time t*

$$\begin{aligned} Q^M(x_t) &= \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} J_k^j(x_k^j) = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} \sum_{i=k}^{T^j} h(x_i^j, u_i^j) \\ &= \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} h(x_k^j, u_k^j) + \sum_{j=0}^{M}\sum_{k=0}^{T^j-1} \lambda_{k|t}^{j,*} J_{k+1}^j(x_{k+1}^j). \end{aligned} \tag{10.14}$$

We notice that the summation of the cost-to-go in the above expression can be rewritten as

$$\sum_{j=0}^{M}\sum_{k=0}^{T^j-1} \lambda_{k|t}^{j,*} J_{k+1}^j(x_{k+1}^j) = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \bar{\lambda}_{k|t}^{j} J_k^j(x_k^j) \geq Q^M(x_{t+1}), \tag{10.15}$$

where $\bar{\lambda}_{k|t}^{j}$ *is the candidate solution defined in* (10.13).

Finally, from equations (10.14) *and* (10.15) *we conclude that the optimal cost is a decreasing Lyapunov function along the closed loop trajectory,*

$$Q^M(x_{t+1}) - Q^M(x_t) \leq -\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} h(x_k^j, u_k^j) < 0, \ \forall \ x_t \in R^n \setminus \{0\}. \tag{10.16}$$

Equation (4.37), the positive definitiveness of $h(\cdot,\cdot)$ *and the continuity of* $Q^M(\cdot)$ *imply that the origin of the closed-loop system* (10.1) *and* (10.10) *is asymptotically stable.*

Proposition 13 *(Cost) Consider the closed-loop system* (10.1) *and* (10.10). *Let Assumptions 17-18 hold and* $\mathcal{CS}^M$ *be the convex safe set defined in* (10.6). *If the initial state* $x_0 \in \mathcal{CS}^M$. *Then, the Q-function at* x_0, $Q^M(x_0)$, *upper bounds the cost associated with the trajectory of closed-loop system* (10.1) *and* (10.10),

$$J\big(x_0\big) = \sum_{k=0}^{\infty} h(x_k, u_k) \leq Q^M(x_0) \tag{10.17}$$

where $\{x_0, \ldots, x_t, \ldots\}$ *and* $\{u_0, \ldots, u_t, \ldots\}$ *are the closed-loop trajectory and associated input sequence, respectively.*

Proof *From* (10.16) *and convexity of* $h(\cdot,\cdot)$, *we have that*

$$Q^M(x_t) \geq h(x_t, u_t) + Q^M(x_{t+1})$$

Using the above equation recursively and from the asymptotic convergence to the origin we have that

$$\begin{aligned} Q^M(x_0) &\geq h(x_0, u_0) + Q^M(x_1) \\ &\geq \sum_{k=0}^{\infty} h(x_k, u_k) + \lim_{k\to\infty} Q^M(x_k) = \sum_{k=0}^{\infty} h(x_k, u_k). \end{aligned}$$

Note that, if the optimal closed-loop trajectory from $x_0 = x_s$ is given, then the approximated value function $Q^M(x_s)$ will be the optimal cost-to-go from x_s. Consequently, *Proposition 3* implies that the proposed data-based policy will behave optimally for $x_0 = x_s$, if the optimal behavior from $x_0 = x_s$ has been observed.

10.4 Examples

In this section we first test the data-based policy (10.8) and (10.10) on a double integrator system. Afterwards, we test the local data-based policy (10.11) and (10.12) on the Berkeley Autonomous Racing Car (BARC) platform.

10.4.1 Example I: Double Integrator

Consider the following discrete time Constrained Linear Quadratic Regulator (CLQR) problem

$$
\begin{aligned}
J^*\big(x_0\big) = \min_{\bar{u}_0, \bar{u}_1, \dots} \quad & \sum_{k=0}^{\infty} \Big[||\bar{x}_k||_2^2 + ||\bar{u}_k||_2^2 \Big] \\
\text{s.t.} \quad & \\
& \bar{x}_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \bar{x}_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \bar{u}_k, \ \forall k \geq 0 \\
& \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \bar{x}_k \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \ \forall k \geq 0 \\
& -1 \leq \bar{u}_k \leq 1 \ \ \forall k \geq 0, \\
& \bar{x}_0 = x_0 = [-1, 3]^\top.
\end{aligned}
\tag{10.18}
$$

First, we construct the convex safe set using one solution to the above CLQR and we empirically validate *Proposition* 1-3. Afterwards, we analyze the effect of the amount of data on the value function approximation and the data-based policy (10.8) and (10.10).

10.4.1.1 Properties verification

First, we compute and store the optimal solution to the CLQR problem (10.18),

$$
\begin{aligned}
&[\bar{x}_0^*, \bar{x}_1^*, \dots, \bar{x}_T^*] \\
&[\bar{u}_0^*, \bar{u}_1^*, \dots, \bar{u}_T^*]
\end{aligned}
\tag{10.19}
$$

where T is the time index at which $||\bar{x}_T^*||_2^2 \leq \epsilon = 10^{-10}$.

The stored optimal trajectory in (10.19) is used to build the convex safe set $\mathcal{CS}^M$ in (10.6) for $M = 1$ and the approximation to the value function $Q^M(\cdot)$ in (10.7). We tested the data-based policy for $x_0 = \bar{x}_0^*$ and for other 10 randomly picked initial conditions inside

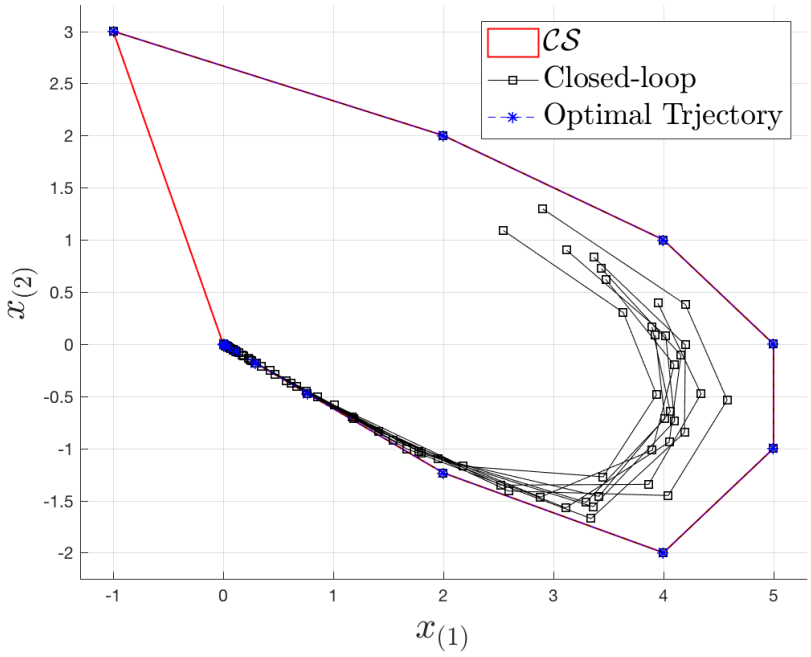


Figure 10.1: Closed-loop trajectories performed by the data-based policy.

$\mathcal{CS}^M$. We denote the resulting closed-loop trajectories and associated input sequences for $j \in \{0, \dots, 9\}$ as

$$\begin{aligned} \mathbf{x}^j &= [x_0^j, \dots, x_{T^j}^j] \\ \mathbf{u}^j &= [u_0^j, \dots, u_{T^j}^j] \end{aligned}. \tag{10.20}$$

Figure 10.1 shows the closed-loop trajectories, we confirm that state and input constraints are satisfies, accordingly to *Proposition 1*. Furthermore, we notice that the closed-loop trajectories converge to the origin as we expected from *Proposition 2*. It is interesting to notice that for $x_0 = \bar{x}_0^*$ the closed-loop trajectory performed by the data-based policy overlaps with the optimal one.

Moreover, we analyze the cost associated with the closed-loop trajectories (10.17). Table 10.1 shows the realized cost (10.17) and the approximated value function $Q^M(\cdot)$ evaluated at different initial conditions. We confirm that $Q^M(x_0)$ upper bounds the performance of the closed-loop trajectory, as shown in *Proposition 3*.

10.4.1.2 The effect of data

Finally, we empirically analyze the effect of data on the Q-function and the data-based policy. First, we construct two approximations to the value function: $Q^{M_1}(\cdot)$ using (10.19) and $M_1 = 10$ stored state and input trajectories computed in the previous subsection (10.20), and $Q^{M_2}(\cdot)$ using (10.19) and the optimal solution to the CLQR for $\bar{x}_0 = [2.9033, 1.2959]$.

Table 10.1: Comparison of the realized cost and value function for different initial conditions

x_0	$J(x_0)$	$Q^M(x_0)$
$[-1, 3]^\top$	112.53	112.53
$[2.9033, 1.2959]^\top$	78.60	89.60
$[3.9495, 0.3921]^\top$	62.00	73.97
$[3.3673, 0.8315]^\top$	66.45	79.23
$[3.4349, 0.7243]^\top$	62.96	76.79
$[3.9253, 0.0874]^\top$	50.37	63.69
$[3.1189, 0.9013]^\top$	63.11	78.18
$[3.8963, 0.1645]^\top$	52.12	65.74
$[2.5449, 1.0898]^\top$	58.04	76.85
$[3.4751, 0.6212]^\top$	59.22	74.06
$[2.5770, 1.1763]^\top$	63.34	80.50

Afterwards, we run the data-based policy using $Q^{M_1}(\cdot)$ and $Q^2(\cdot)$. Table 10.2 shows the cost associated with the closed-loop trajectories $J^i(\cdot)$ and the value function approximation $Q^i(\cdot)$, for $i = \{1, 2\}$. We notice that $Q^{M_1}(x_0)$ lower bounds $Q^M(x_0)$ from Table 10.1 and, therefore, better approximates the value function. However, the realized cost $J^1(x_0)$ does not improve with respect to $J(x_0)$ from Table 10.1. On the other hand, we notice that the data-based policy constructed using $Q^2(\cdot)$ is able to improve the closed-loop performance $J^2(x_0)$. It is interesting to notice that $Q^{M_1}(x_0)$ is constructed using one optimal trajectory and 10 feasible trajectories, whereas $Q^2(x_0)$ is constructed using just two optimal trajectories. This result is interesting and it suggests that not all data points are equally valuable.

Table 10.2: Comparison of the realized cost and value function for different initial conditions

x_0	$J^1(x_0)$	$Q^{M_1}(x_0)$	$J^2(x_0)$	$Q^{M_2}(x_0)$
$[-1, 3]^\top$	112.53	112.53	112.53	112.53
$[2.9033, 1.2959]^\top$	78.60	78.60	72.89	72.89
$[3.9495, 0.3921]^\top$	62.00	62.00	59.43	62.12
$[3.3673, 0.8315]^\top$	66.45	66.45	61.86	66.39
$[3.4349, 0.7243]^\top$	62.96	62.96	58.97	64.38
$[3.9253, 0.0874]^\top$	50.37	50.37	49.24	54.57
$[3.1189, 0.9013]^\top$	63.11	63.11	58.76	65.04
$[3.8963, 0.1645]^\top$	52.12	52.12	50.73	55.86
$[2.5449, 1.0898]^\top$	58.04	58.04	53.85	62.65
$[3.4751, 0.6212]^\top$	59.22	59.22	55.81	62.12
$[2.5770, 1.1763]^\top$	63.34	63.34	58.63	65.74

10.4.2 Example II: Autonomous Racing Experiments

In this section, we test the proposed control strategy on a 1/10-scale open source vehicle platform called the Berkeley Autonomous Race Car (BARC)[1]. The BARC is equipped with an inertial measurement unit, encoders, and an ultrasound-based indoor GPS system. The vehicle has an Odroid XU4 which is used for collecting data and running the state estimator. Finally, the computation are performed on a MSI laptop with an intel CORE i7. A video of the experiments can be found here: `https://youtu.be/pB2pTedXLpI`.

The control task is to drive the vehicle continuously around the track minimizing the lap time, while being within the track boundaries. The state vector is

$$x = [v_x, v_y, w_z, e_\psi, s, e_y]^\top$$

where v_x, v_y and w_z represent the vehicle's longitudinal, lateral and angular velocity in the body fixed frame. The position of the system is measured with respect to the curvilinear reference frame [83], where s represents the progress of the vehicle along the centerline of the track, e_ψ and e_y represent the heading angle and lateral distance error between the vehicle and the path. It is important to underline that, given the lane boundaries $e_{y_{min}}$ and $e_{y_{max}}$, the feasible region $\mathcal{X} = \{x \in \mathbb{R}^n : e_{y_{min}} \leq e_6^\top x \leq e_{y_{max}}\}$ for $e_6 = [0, 0, 0, 0, 0, 1]^\top$ is a convex set. The control input vector is $u = [\delta, a]$ where δ and a are the steering angle and acceleration, respectively. The input constraints are

$$\begin{aligned} -0.25[\text{rad}] \leq &\delta \leq 0.25[\text{rad}] \\ -0.7[\text{m/s}^2] \leq &a \leq 2[\text{m/s}^2]. \end{aligned}$$

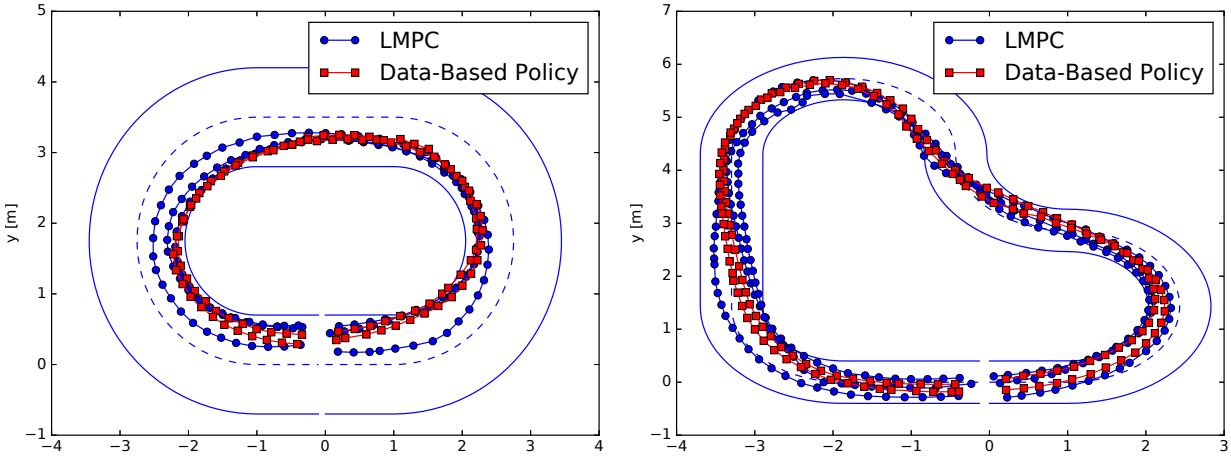
Finally, we underline that the autonomous racing problem is a repetitive task and the goal is not to steer the system to the origin. Therefore, we use the method from Chapter 9 to apply the proposed strategy to the autonomous racing repetitive control problem. In particular, we define the set of state beyond the finish line of the track of length L, $\mathcal{X}_F = \{x \in \mathbb{R}^6 : e_5^\top x \geq L\}$ and we use the set $\mathcal{X}_F$ to compute the cost associated with the stored trajectories

$$h(x, u) = \begin{cases} 1 & \text{If } x \notin \mathcal{X}_F \\ 0 & \text{If } x \in \mathcal{X}_F \end{cases}.$$

For the first 29 laps of the experiment, we run the Learning Model Predictive Controller (LMPC) from Chapter 9 to learn a fast trajectory which drives the vehicle around the track. From the 30th lap, we run the local data-based policy (10.11) and (10.12) using the latest $M = 8$ laps and $N = 10$ stored data for each lap. Therefore, the control action is computed upon solving the small optimization problem (10.11) where $[\lambda^0_{0|t}, \dots, \lambda^j_{k|t}, \dots, \lambda^M_{T^M|t}] \in \mathbb{R}^{M|\mathcal{K}^j(x)|}$ with $M|\mathcal{K}^j(x)| = 80$.

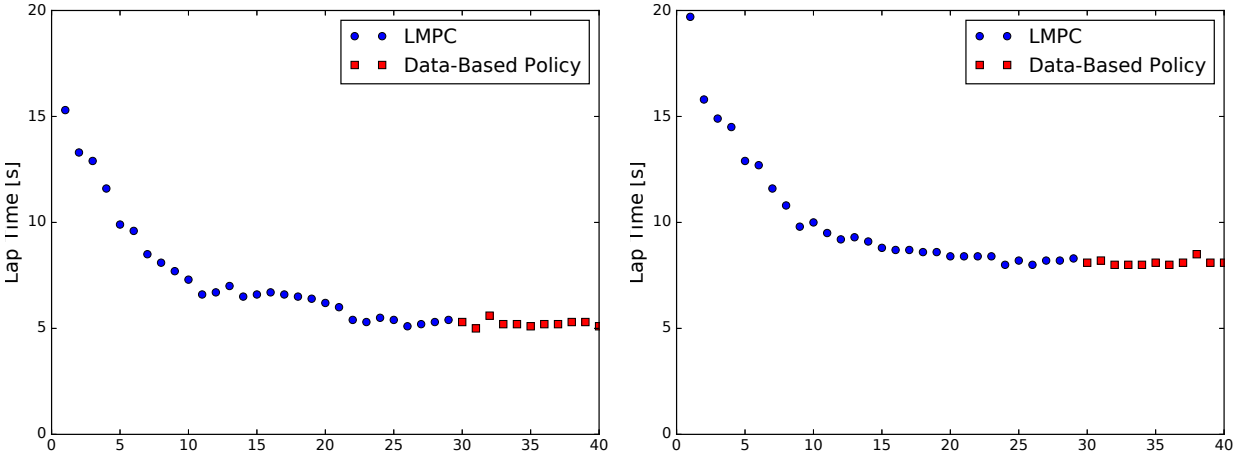
We tested the controller on an oval-shaped and L-shaped tracks. Figures 10.2, 10.4 and 10.5 show that the local data-based policy (10.11) and (10.12) is able to drive the

[1]More information at the project site barc-project.com



(a) Oval-shaped track. (b) L-shaped track.

Figure 10.2: In red squares are shown the closed-loop trajectories performed by the data-based policy on the oval-shaped track. In blue circles are reported three trajectories in the sampled safe set. Finally, the green dashed line marks the centerline of the track.



(a) Oval-shaped track. (b) L-shaped track.

Figure 10.3: Lap time on L-shaped track over the lap number. At the 30th lap the data-based policy drives the vehicle around the track without worsening the closed loop-performance.

vehicle around the track satisfying input and state constraints. Furthermore, we notice that the closed-loop trajectories generated with the local data-based policy lies in the convex hull of the sampled safe set $\mathcal{SS}$, which is constructed from the last 8 trajectories performed by the LMPC. It is interesting to notice that the real system is nonlinear but smooth and, for this reason, the system dynamics can be locally linearized. Intuitively, the existence of a local linear model allows us to use the local data-based policy to safely drive the vehicle. Indeed at each time t the controller uses only the historical data close to the system's state x_t.

Figure 10.3 report the lap time over the lap number. We notice that the data-based policy is able to safely drive the vehicle around the track, without hurting the closed-loop performance. In particular, the data-based policy is able to replicate the best lap times performed by the LMPC controller on both tracks.

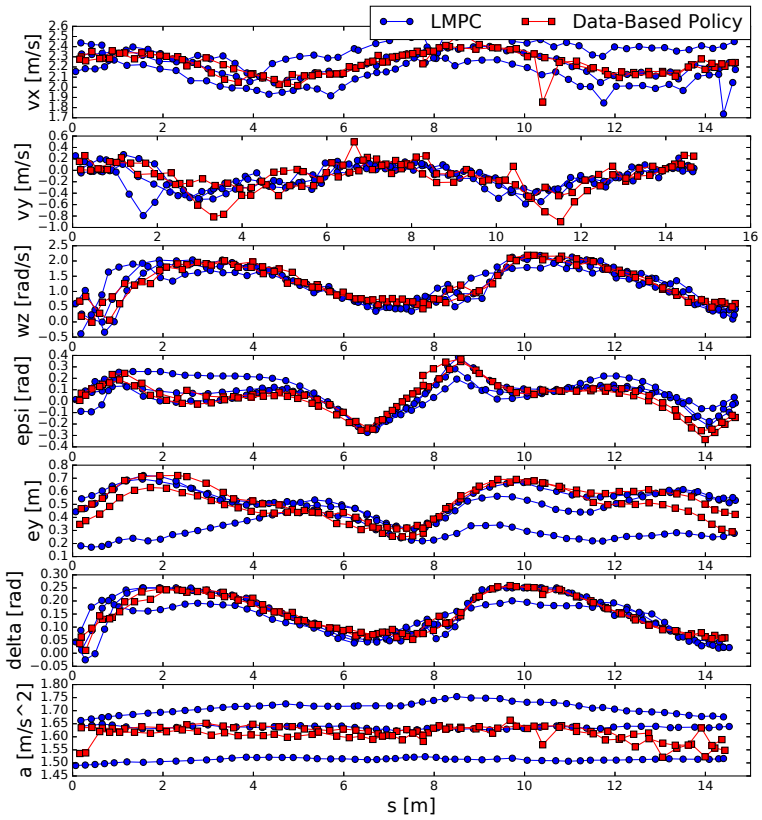


Figure 10.4: Closed-loop trajectory and associated inputs of the data-based policy and LMPC on the oval-shaped track.

Finally, we analyze the computational time. We compare the computational cost as-

sociated with the proposed data-based policy and with the LMPC. Table 10.3 shows that on average it took $\sim$ 1.3ms to evaluate the proposed data-based policy and $\sim$ 29.5ms to evaluate the LMPC policy.

Table 10.3: Comparison of computational time

	Avarage	Min	Max	Std Deviation
LMPC	29.5ms	21.8ms	50.0ms	6.1ms
Data-Based Policy	1.3ms	1.1ms	2.3ms	0.2ms

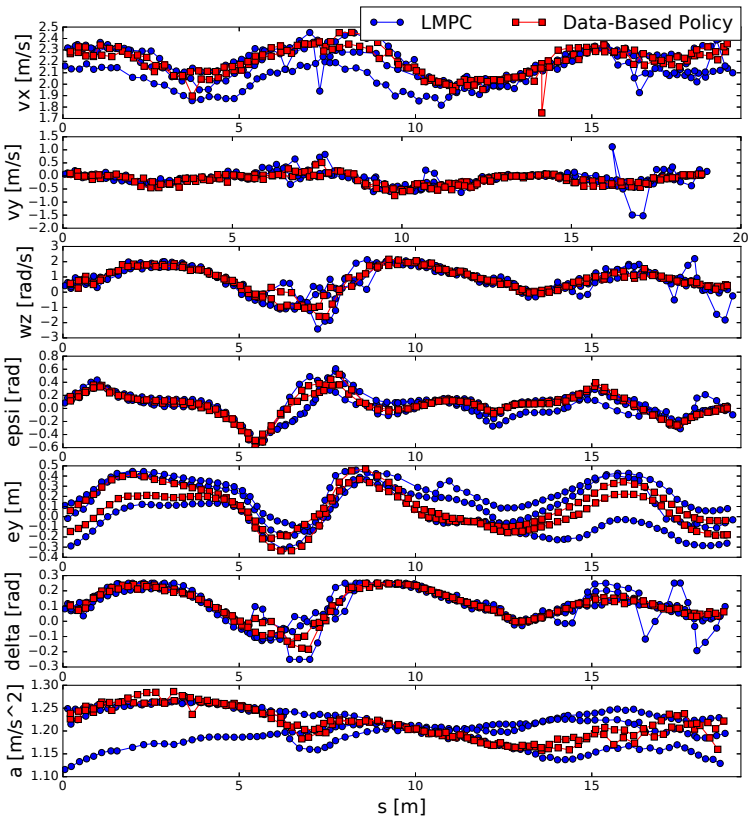


Figure 10.5: Closed-loop trajectory and associated inputs of the data-based policy and LMPC on the L-shaped track.

10.5 Appendix

In this Appendix, we show that the proposed data-based policy may be used to steer a linear time invariant system to a terminal invariant set $\mathcal{X}_F$. In order to prove that the properties from Propositions 1-3 hold also in this settings the following assumptions must hold.

Assumption 19 *The terminal set $\mathcal{X}_F$ is defined by the convex hull of the terminal state of the stored trajectories* (10.4)*, i.e. $\mathcal{X}_F = Conv\big(\cup_{j=0}^{M} x_{T^j}^j\big)$.*

Assumption 20 *All $M+1$ input and state sequences in* (10.3)-(10.4) *are feasible and known. Furthermore, assume that the state sequence in* (10.4) *converges to the terminal set $\mathcal{X}_F$ and the terminal input $u_{T^j}^j$ keeps the evolution of the system* (10.1) *into $\mathcal{X}_F$. More formally, we assume that $x_{T^j}^j \in \mathcal{X}_F, \forall j \in \{0,\dots,M\}$ and $Ax_{T^j} + Bu_{T^j} \in \mathcal{X}_F$.*

Proposition 14 *(Feasibility) Consider the closed-loop system* (10.1) *and* (10.10)*. Let Assumptions 19-20 hold and $\mathcal{CS}^M$ be the convex safe set defined in* (10.6)*. If the initial state $x_0 \in \mathcal{CS}^M$. Then, the data-based policy* (10.8) *and* (10.10) *is feasible for all time $t \geq 0$.*

Proof *The proof follows from linearity of the system.*
We assume that at time t the system state $x_t \in \mathcal{CS}^M$, therefore the optimization problem (10.8) *is feasible. Let* (10.9) *be the optimal solution to* (10.8)*, then at the next time step $t+1$ we have*

$$\begin{aligned} x_{t+1} &= Ax_t + B\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} u_k^j \\ &= A\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} x_k^j + B\sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*} u_k^j \\ &= \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*}(Ax_k^j + Bu_k^j) \in \mathcal{CS}^M. \end{aligned}$$

By Assumption 20 we have that for all $\forall j \in \{0,\dots M\}$ it exist $\lambda_k^j \geq 0$ such that $\sum_{k=0}^{M} \lambda_k^j = 1$ and

$$\begin{aligned} \sum_{j=0}^{M} \lambda_{T^j|t}^{j,*}(Ax_{T^j}^j + Bu_{T^j}^j) &= \sum_{j=0}^{M} \lambda_{T^j|t}^{j,*} \sum_{k=0}^{M} \lambda_k^j x_{T^k}^k \\ &= \sum_{k=0}^{M}\sum_{j=0}^{M} \lambda_{T^j|t}^{j,*} \lambda_k^j x_{T^k}^k = \sum_{k=0}^{M} \tilde{\lambda}_k x_{T^k}^k \end{aligned}$$

where $\forall k \in \{0,\dots M\}$ we defined $\tilde{\lambda}_k = \sum_{i=0}^{M} \lambda_{T^i|t}^{i,*} \lambda_k^i$. It follows that

$$x_{t+1} = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \lambda_{k|t}^{j,*}(Ax_k^j + Bu_k^j) = \sum_{j=0}^{M}\sum_{k=0}^{T^j} \bar{\lambda}_k^j x_k^j$$

where $\forall j \in \{0, \dots M\}$

$$\begin{aligned} &\bar{\lambda}^j_0 = 0, \\ &\bar{\lambda}^j_{k_j} = \lambda^{j,*}_{k_j-1|t}, \qquad\qquad \forall k_j \in \{1, \dots, T^j - 1\} \\ &\bar{\lambda}^j_{T^j} = \lambda^{j,*}_{T^j-1|t} + \tilde{\lambda}_j \end{aligned} \tag{10.21}$$

is a feasible solution to the optimization problem (10.8) *at time* $t+1$.
By assumption we have at time $t = 0$ *the state* $x_0 \in \mathcal{CS}^M$. *Furthermore, we have shown that if at time* t *the state* $x_t \in \mathcal{CS}^M$, *then at time* $t+1$ *the state* $x_{t+1} \in \mathcal{CS}^M$ *and the optimization problem* (10.8) *is feasible. Therefore by induction we conclude that* $x_t \in \mathcal{CS}^M \subseteq \mathcal{X}$, $\forall t \in \mathbb{Z}_{0+}$ *and that the optimization problem* (10.8) *is feasible* $\forall t \in \mathbb{Z}_{0+}$.

In order to prove convergence we make the following assumption on the stage cost.

Assumption 21 *The stage cost* $h(\cdot, \cdot)$ *is a continuous convex function and* $\forall u \in \mathcal{U}$ *it satisfies*

$$h(x, u) = 0, \forall x \in \mathcal{X}_F \text{ and } h(x, u) \succ 0 \ \forall \ x \in \mathbb{R}^n \setminus \{\mathcal{X}_F\}.$$

Proposition 15 *(Convergence) Consider the closed-loop system* (10.1) *and* (10.10). *Let Assumption 19-21 hold and* $\mathcal{CS}^M$ *be the convex safe set defined in* (10.6). *If the initial state* $x_0 \in \mathcal{CS}^M$. *Then, the origin of the closed-loop system* (10.1) *and* (10.10) *is asymptotically stable.*

Proof *The proof follows from the proof of Proposition 2. In particular, the candidate solution* (10.21) *may be exploited to show that* $Q^M(\cdot)$ *is Lyapunov function along the closed-loop trajectory.*

Proposition 16 *(Cost) Consider the closed-loop system* (10.1) *and* (10.10). *Let Assumptions 19-21 hold and* $\mathcal{CS}^M$ *be the convex safe set defined in* (10.6). *If the initial state* $x_0 \in \mathcal{CS}^M$. *Then, the Q-function at* x_0, $Q^M(x_0)$, *upper bounds the cost associated with the trajectory of closed-loop system* (10.1) *and* (10.10),

$$J\big(x_0\big) = \sum_{k=0}^{\infty} h(x_k, u_k) \leq Q^M\big(x_0\big)$$

where $\{x_0, \dots, x_t, \dots\}$ *and* $\{u_0, \dots, u_t, \dots\}$ *are the closed-loop trajectory and associated input sequence, respectively.*

Proof *The proof follows as in Proposition 13.*

Chapter 11

Conclusions

In this thesis, we presented the Learning Model Predictive Control (LMPC) framework. We proposed to iteratively synthesize control policies by exploiting historical data from autonomous systems performing iterative tasks. First, we introduced the LMPC design for deterministic systems, which guarantee recursive constraint satisfaction, closed-loop stability and performance improvement. Furthermore we show that, under mild assumptions, if the policy update process has converged (i.e. using new closed-loop data in the synthesis process does not change the control policy), then the closed-loop behavior is optimal for the entire task. Afterwards, we described the control design for uncertain systems. Finally, we proposed a system identification strategy for iterative tasks, and we tested the controller on the Berkeley Autonomous Race Car (BARC) platform. Experimental results showed that the LMPC learns to safety drive a vehicle at the limits of handling. Moreover, we showed that when the controller has converged to a steady-state behavior, the closed-loop data may be used to synthesize a model-free policy which allows us to reduce the computational burden.

11.1 Future Work

In the following, we outline several possible extensions for future works.

Output Feedback

The proposed framework assumes perfect state measurements. This assumption may be weakened if we consider a linear system

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t \\ y_t &= Cx_t \end{aligned}$$

where (A, C) is an observable pair. In this case, it would be possible to design an output feedback LMPC, where the safe set and value function approximation are defined over the output space.

System Identification

In Chapter 9, we proposed a system identification strategy for iterative tasks. We constructed

an affine time varying model around a candidate trajectory. In particular, given the candidate state-input trajectory

$$[(\bar{x}_0, \bar{u}_0), \dots, (\bar{x}_N, \bar{u}_N)], \tag{11.1}$$

we identified an affine time varying model $x_{t+1} = A_t x_t + B_t u_t + C_t$ around $(\bar{x}_t, \bar{u}_t)$. When a nonlinear model $x_{t+1} = f(x_t, u_t)$ is approximated using this strategy, the prediction accuracy depends on the distance between the state-input pair (x_t, u_t) and the candidate pair $(\bar{x}_t, \bar{u}_t)$. Basically, the prediction error is low in a the neighborhood of $(\bar{x}_t, \bar{u}_t)$. Therefore, in order to reduce the prediction error over the planning horizon, we could constraint the predicted LMPC trajectory to lie close to the candidate one from (11.1). This strategy would ensure safety and performance improvement with some confidence, when the system model is estimated from data.

Exploration and Adaptation

As we have discussed in this thesis, the LMPC iteratively explores the state space to improve the closed-loop performance. This exploration process is model-based and cost driven. Indeed, at each time step, the controller plans a trajectory which minimizes the predicted cost over a moving horizon. It would be interesting to explore a variation to this strategy, where the controller minimizes both the predicted cost and a tracking cost with respect to the best stored trajectory. Ideally, the tracking cost would be tuned online as a function of the historical and current prediction error. This strategy would result in a cautious controller, which explores the state space if the prediction error is small (i.e., if the model is good enough for control) and, otherwise, tracks the best safe stored trajectory.

Stochastic Analysis

In Chapter 6, we proposed to use roll-outs of the closed-loop uncertain system to construct the LMPC policy. We showed that when roll-outs are used in the synthesis process, the properties of the safe set and value function approximation are satisfied with some ϵ-probability. There are two questions which arise. How many roll-outs are required to achieve a specific ϵ-level of probability? What is the probability of failure for the closed-loop system? Answering these questions would allow us to choose a priori the number of roll-outs needed to perform the LMPC policy update.

Learning From Failure

Throughout this thesis, we have discussed how to iteratively improve the closed-loop performance while guaranteeing safety. We have shown that learning while being safe is possible when a model of the system, disturbance and environment are given. When these quantities are inaccurate, the closed-loop system may violate the safety constraints. In this scenario, we want to detect failure as early as possible. For instance, in autonomous driving we want to understand when a collision is inevitable to reduce the harm. Detecting failures may be possible by online monitoring of the closed-loop cost, which should be a Lyapunov function. Indeed, when the closed-loop cost is not decreasing, the closed-loop system has not evolved as planned. Finally, when failure occurs, we would like to leverage this experience. I believe that unsafe trajectories carry important information (i.e., unsafe regions of the state-input space), and how to use them in the synthesis process is an interesting open question.

Hybrid Systems

In this thesis, we have shown that for linear systems and for a class of nonlinear systems, the safe set and value function approximation may be convexified. The resulting relaxed LMPC formulation allows us to guarantee safety and performance improvement, while decreasing the computation burden. It would be interesting to investigate LMPC relaxation strategies for hybrid systems. It is well-known that trajectory planning for such systems is computationally expensive, as the modes of operation are described by integer variables. However in iterative tasks, we may use historical data to fix different sequences of operational modes. This strategy would allows us to reduce the computational burden while safely exploring the state space to improve the closed-loop performance.

Bibliography

[1] Richard M Murray et al. "Future directions in control in an information-rich world". In: *IEEE Control Systems Magazine* 23.2 (2003), pp. 20–33.

[2] *Autonomous and ADAS test cars produce over 11 TB of data per day*. `https://www.tuxera.com/blog/autonomous-and-adas-test-cars-produce-over-11-tb-of-data-per-day/`. Accessed: 2019-11-3.

[3] Anil Aswani et al. "Provably safe and robust learning-based model predictive control". In: *Automatica* 49.5 (2013), pp. 1216–1226.

[4] Juš Kocijan et al. "Gaussian process model based predictive control". In: *Proceedings of the 2004 American Control Conference*. Vol. 3. IEEE. 2004, pp. 2214–2219.

[5] Torsten Koller et al. "Learning-Based Model Predictive Control for Safe Exploration". In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE. 2018, pp. 6059–6066.

[6] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. "Safe controller optimization for quadrotors with Gaussian processes". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 491–496.

[7] Lukas Hewing, Alexander Liniger, and Melanie N Zeilinger. "Cautious NMPC with gaussian process dynamics for autonomous miniature race cars". In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 1341–1348.

[8] Enrico Terzi et al. "Learning multi-step prediction models for receding horizon control". In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 1335–1340.

[9] Sarah Dean et al. "Safely learning to control the constrained linear quadratic regulator". In: *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 5582–5588.

[10] F. Borrelli, A. Bemporad, and M. Morari. *Predictive Control for linear and hybrid systems*. Cambridge University Press, 2017.

[11] 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.

[12] Benjamin Recht. "A tour of reinforcement learning: The view from continuous control". In: *Annual Review of Control, Robotics, and Autonomous Systems* (2018).

[13] Dimitri P Bertsekas. "Biased Aggregation, Rollout, and Enhanced Policy Improvement for Reinforcement Learning". In: *Lab. for Information and Decision Systems Report, MIT* (2018).

[14] Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*. Vol. 5. Athena Scientific Belmont, MA, 1996.

[15] Francesco Borrelli. *Constrained optimal control of linear and hybrid systems*. Vol. 290. Springer, 2003.

[16] Elmer G Gilbert and K Tin Tan. "Linear systems with state and control constraints: The theory and application of maximal output admissible sets". In: *IEEE Transactions on Automatic control* 36.9 (1991), pp. 1008–1020.

[17] Yuandan Lin, Eduardo Sontag, and Yuan Wang. "Various results concerning set input-to-state stability". In: *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*. Vol. 2. IEEE. 1995, pp. 1330–1335.

[18] Zhong-Ping Jiang and Yuan Wang. "Input-to-state stability for discrete-time nonlinear systems". In: *Automatica* 37.6 (2001), pp. 857–869.

[19] 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.

[20] Hassan K Khalil and JW Grizzle. *Nonlinear systems*. Vol. 3. Prentice hall Upper Saddle River, NJ, 2002.

[21] Lars Grüne and Christopher M Kellett. "ISS-Lyapunov functions for discontinuous discrete-time systems". In: *IEEE Transactions on Automatic Control* 59.11 (2014), pp. 3098–3103.

[22] Christopher M Kellett. "A compendium of comparison function results". In: *Mathematics of Control, Signals, and Systems* 26.3 (2014), pp. 339–374.

[23] Carlos E Garcia, David M Prett, and Manfred Morari. "Model predictive control: theory and practice-a survey". In: *Automatica* 25.3 (1989), pp. 335–348.

[24] Manfred Morari and Jay H Lee. "Model predictive control: past, present and future". In: *Computers & Chemical Engineering* 23.4 (1999), pp. 667–682.

[25] David Q Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814.

[26] J. Rawlings and D. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2009.

[27] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer Verlag, 2013.

[28] Kurtland Chua et al. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models". In: *Advances in Neural Information Processing Systems*. 2018, pp. 4759–4770.

[29] Ugo Rosolia and Francesco Borrelli. "Learning how to autonomously race a car: a predictive control approach". In: *IEEE Transactions on Control Systems Technology* (2019).

[30] Kwang Soon Lee and Jay H Lee. "Model predictive control for nonlinear batch processes with asymptotically perfect tracking". In: *Computers & Chemical Engineering* 21 (1997), S873–S879.

[31] Chris J Ostafew, Angela P Schoellig, and Timothy D Barfoot. "Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 4029–4036.

[32] Kwang S Lee and Jay H Lee. "Convergence of constrained model-based predictive control for batch processes". In: *IEEE Transactions on Automatic Control* 45.10 (2000), pp. 1928–1932.

[33] Jay H Lee, Kwang S Lee, and Won C Kim. "Model-based iterative learning control with a quadratic criterion for time-varying linear systems". In: *Automatica* 36.5 (2000), pp. 641–657.

[34] Richard S Sutton. "Learning to predict by the methods of temporal differences". In: *Machine learning* 3.1 (1988), pp. 9–44.

[35] Steven J Bradtke and Andrew G Barto. "Linear least-squares algorithms for temporal difference learning". In: *Machine learning* 22.1-3 (1996), pp. 33–57.

[36] Marko Bacic et al. "General interpolation in MPC and its advantages". In: *IEEE Transactions on Automatic Control* 48.6 (2003), pp. 1092–1096.

[37] Florian D Brunner, Mircea Lazar, and Frank Allgöwer. "Stabilizing linear model predictive control: On the enlargement of the terminal set". In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 511–517.

[38] Chris J Ostafew, Angela P Schoellig, and Timothy D Barfoot. "Robust constrained learning-based NMPC enabling reliable mobile robot path tracking". In: *The International Journal of Robotics Research* 35.13 (2016), pp. 1547–1563.

[39] Xiaonan Lu and Mark Cannon. "Robust adaptive tube model predictive control". In: *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 3695–3701.

[40] Matthias Lorenzen, Mark Cannon, and Frank Allgöwer. "Robust MPC with recursive model update". In: *Automatica* 103 (2019), pp. 461–471.

[41] Matthias Lorenzen, Frank Allgöwer, and Mark Cannon. "Adaptive model predictive control with robust constraint satisfaction". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 3313–3318.

[42] Marko Tanaskovic et al. "Adaptive model predictive control for constrained linear systems". In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 382–387.

[43] Marko Tanaskovic et al. "Adaptive receding horizon control for constrained MIMO systems". In: *Automatica* 50.12 (2014), pp. 3019–3029.

[44] Monimoy Bujarbaruah et al. "Adaptive MPC for Iterative Tasks". In: *2018 IEEE Conference on Decision and Control (CDC)* (2018), pp. 6322–6327.

[45] Monimoy Bujarbaruah, Xiaojing Zhang, and Francesco Borrelli. "Adaptive MPC with Chance Constraints for FIR Systems". In: *2018 Annual American Control Conference (ACC)*. June 2018, pp. 2312–2317.

[46] Monimoy Bujarbaruah et al. "Adaptive MPC under time varying uncertainty: Robust and Stochastic". In: *arXiv preprint arXiv:1909.13473* (2019).

[47] Alberto Bemporad and Manfred Morari. "Robust model predictive control: A survey". In: *Robustness in identification and control*. Springer, 1999, pp. 207–226.

[48] 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.

[49] Martin Evans, Mark Cannon, and Basil Kouvaritakis. "Robust and stochastic linear MPC for systems subject to multiplicative uncertainty". In: *IFAC Proceedings Volumes* 45.17 (2012), pp. 335–341.

[50] Felix Berkenkamp and Angela P Schoellig. "Safe and robust learning control with Gaussian processes". In: *2015 European Control Conference (ECC)*. IEEE. 2015, pp. 2496–2501.

[51] Enrico Terzi et al. "Learning-based predictive control for linear systems: a unitary approach". In: *Automatica* 108 (2019), p. 108473.

[52] Enrico Terzi et al. "Robust predictive control with data-based multi-step prediction models". In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 1710–1715.

[53] Sarah Dean et al. "Regret bounds for robust adaptive control of the linear quadratic regulator". In: *Advances in Neural Information Processing Systems*. 2018, pp. 4188–4197.

[54] Yuh-Shyang Wang, Nikolai Matni, and John C Doyle. "A system level approach to controller synthesis". In: *IEEE Transactions on Automatic Control* (2019).

[55] Jaime F Fisac et al. "A general safety framework for learning-based control in uncertain robotic systems". In: *IEEE Transactions on Automatic Control* (2018).

[56] Shahab Kaynama et al. "The continual reachability set and its computation using maximal reachability techniques". In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE. 2011, pp. 6110–6115.

[57] John Lygeros, Claire Tomlin, and Shankar Sastry. "Controllers for reachability specifications for hybrid systems". In: *Automatica* 35.3 (1999), pp. 349–370.

[58] Alexander Liniger and John Lygeros. "Real-time control for autonomous racing based on viability theory". In: *IEEE Transactions on Control Systems Technology* 99 (2017), pp. 1–15.

[59] Kim P Wabersich and Melanie N Zeilinger. "Linear model predictive safety certification for learning-based control". In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE. 2018, pp. 7130–7135.

[60] Franco Blanchini and Felice Andrea Pellegrino. "Relatively optimal control and its linear implementation". In: *IEEE Transactions on Automatic Control* 48.12 (2003), pp. 2151–2162.

[61] 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.

[62] Ugo Rosolia and Francesco Borrelli. "Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework." In: *IEEE Transactions on Automatic Control* (2017).

[63] Colin N Jones and Manfred Morari. "Polytopic approximation of explicit model predictive controllers". In: *IEEE Transactions on Automatic Control* 55.11 (2010), pp. 2542–2553.

[64] Steven Diamond and Stephen Boyd. "CVXPY: A Python-Embedded Modeling Language for Convex Optimization". In: *Journal of Machine Learning Research* 17.83 (2016), pp. 1–5.

[65] Lester E Dubins. "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents". In: *American Journal of mathematics* 79.3 (1957), pp. 497–516.

[66] Yiqi Gao et al. "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads". In: *ASME 2010 dynamic systems and control conference*. American Society of Mechanical Engineers. 2010, pp. 265–272.

[67] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[68] Francesco Bullo et al. "Dynamic vehicle routing for robotic systems". In: *Proceedings of the IEEE* 99.9 (2011), pp. 1482–1504.

[69] Yoshiaki Kuwata et al. "Motion planning for urban driving using RRT". In: *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 1681–1686.

[70] Sertac Karaman and Emilio Frazzoli. "Incremental sampling-based algorithms for optimal motion planning". In: *Robotics Science and Systems VI* 104 (2010).

[71] Hans Pirnay, Rodrigo López-Negrete, and Lorenz T Biegler. "Optimal sensitivity based on IPOPT". In: *Mathematical Programming Computation* 4.4 (2012), pp. 307–331.

[72] Daniel Liberzon. *Calculus of variations and optimal control theory: a concise introduction*. Princeton University Press, 2011.

[73] David Q Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814.

[74] Irving Bogner and F Louis Kazda. "An investigation of the switching criteria for higher order contactor servomechanisms". In: *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry* 73.3 (1954), pp. 118–127.

[75] Richard Bellman, Irving Glicksberg, and Oliver Gross. "On the bang-bang control problem". In: *Quarterly of Applied Mathematics* 14.1 (1956), pp. 11–18.

[76] Revaz Valer'yanovich Gamkrelidze. *On the theory of optimal processes in linear systems*. Tech. rep. Joint Publications Research Service Arlington VA, 1961.

[77] Joseph P LaSalle. "The time optimal control problem". In: *Contributions to the theory of nonlinear oscillations* 5 (1959), pp. 1–24.

[78] VG Boltyanskiy, Revaz V Gamkrelidze, and LS Pontryagin. *Theory of optimal processes*. Tech. rep. Joint Publications Research Service Arlington VA, 1961.

[79] 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.

[80] Pierre OM Scokaert and DQ Mayne. "Min-max feedback model predictive control for constrained linear systems". In: *IEEE Transactions on Automatic control* 43.8 (1998), pp. 1136–1142.

[81] Alberto Bemporad, Francesco Borrelli, and Manfred Morari. "Min-max control of constrained uncertain discrete-time linear systems". In: *IEEE Transactions on automatic control* 48.9 (2003), pp. 1600–1606.

[82] Basil Kouvaritakis and Mark Cannon. *Model Predictive Control: Classical, Robust and Stochastic*. Springer, 2015.

[83] Alain Micaelli and Claude Samson. "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots". PhD thesis. INRIA, 1993.

[84] Vassiliy A Epanechnikov. "Non-parametric estimation of a multivariate probability density". In: *Theory of Probability & Its Applications* 14.1 (1969), pp. 153–158.

[85] Bartolomeo Stellato et al. "OSQP: An operator splitting solver for quadratic programs". In: *2018 UKACC 12th International Conference on Control (CONTROL)*. IEEE. 2018, pp. 339–339.

[86] Maximilian Brunner et al. "Repetitive learning model predictive control: An autonomous racing example". In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE. 2017, pp. 2545–2550.

[87] Paul A Theodosis and J Christian Gerdes. "Nonlinear optimization of a racing line for an autonomous racecar using professional driving techniques". In: *ASME 2012 5th Annual Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2012, pp. 235–241.