Receding Horizon Control with Sliding Surfaces and its Application to Vehicle Dynamics

by

Andreas Hansen

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

Fall 2017

Receding Horizon Control with Sliding Surfaces and its Application to Vehicle Dynamics

Abstract

Receding Horizon Control with Sliding Surfaces and its Application to Vehicle Dynamics

by

Andreas Hansen

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In light of the current surge in mobile robotics development, receding horizon control has become of high interest to both academia and industry. This control technique is a systematic way to compute intelligent control actions while accounting for complicated system dynamics, forecasted environment information, and system constraints. This work contributes to the field of receding horizon control by merging sliding surfaces into the controller design process. The resulting methodology is named receding horizon sliding control. It is shown that the invariance properties of sliding surfaces can be exploited for deriving provably stable and persistently feasible controllers for a wide class of constrained nonlinear systems. Furthermore, for linear systems this work reduces the complexity of receding horizon tracking controllers compared to current state-of-the-art methods by exploiting the flatness of sliding hyperplanes. The practicality of the proposed control approach is demonstrated by using applications from vehicle dynamics. The applications include an autonomous underwater robotics problem, an automotive engine control scenario, and a self-driving vehicle control case study. Simulations and real-world experiments confirm the effectiveness of the developed control methodology. The results indicate that the proposed control scheme is typically easy to tune, behaves well under system uncertainty, and has manageable computational requirements that make it amenable to fast systems with sampling times of the order of fractions of seconds.

To J. Karl Hedrick,

The German term for Ph.D. advisor translates to doctoral father, which succinctly describes your ease and composure, your cheerfulness and enduring support, as well as your eagerness to train me as an engineer. I am proud to be a part of your academic family and I am deeply grateful for the many life lessons that I learned from you beyond engineering. You tremendously inspired me and I strive to inspire others with the balance of intellectual excellence and thoughtful leadership that I inherited from you.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AUV	autonomous underwater vehicle
CAD	computer-aided design
CFTOC	constrained finite-time optimal control
COG	center of gravity
CPU	central processing unit
DOF	degree of freedom
ECU	engine control unit
GPS	global positioning system
IMU	inertial measurement unit
LIDAR	light detection and ranging
LOS	line of sight
MIMO	multi-input multi-output
MPC	model predictive control
MPT3	multi-parametric toolbox 3
P	proportional
PD	proportional-derivative
PI	proportional-integral
RAM	random-access memory
RHSC	receding horizon sliding control
RTK	real time kinematic

SC sliding control

SISO single-input single-output

SLAM simultaneous localization and mapping

SMC sliding mode control

SQP sequential quadratic programming

UKF unscented Kalman filter

Acknowledgments

Firstly, I would like to acknowledge my dissertation chair, Francesco Borrelli. I am deeply thankful for his work as co-director of the Vehicle Dynamics and Control Laboratory, also known as VDL. Our group in general and myself in particular greatly benefited from his tireless support and guidance. His sharp intellect is a constant source of inspiration and I am grateful for the opportunity to learn from his wealth of knowledge and expertise.

I would like to thank my dissertation committee members, Masayoshi Tomizuka and Pieter Abbeel, for their service on my committee and for greatly inspiring me in extending the horizon of my scientific work and perspective. My academic journey that started at Hamburg University of Technology and now concludes at the University of California would not have been possible without Edwin Kreuzer, George Leitmann, Oliver O'Reilly, Hanna Peywand Kiani, Wolfgang Mackens, Heinz Herwig, and Paige Lee – thanks to all for their work, their support, and their confidence in my potential.

I gratefully acknowledge the financial support received from the German Academic Exchange Service or DAAD, the Toyota Motor Corporation, and the Hyundai Motor Company. In particular, I want to thank Ken Butts of Toyota for many years of productive collaboration. Also, my thanks go to Mahdi Shahbakhti and Chan Kyu Lee for their enduring support and collaborative spirit. I still remember the first time I met the VDL crew – thanks to Mark Godwin, Brandon Basso, Andrew Gray, Sanghyun Hong, Shih-Yuan Liu, Selina Pan, and Jared Garvey for creating an incredibly welcoming atmosphere for me as a new group member. Furthermore, I would like to thank my fantastic current and past colleagues including Kyle Edelberg, Yi-Wen Liao, Ashwin Carvalho, Jason Kong, Yongkeun Choi, Donghan Lee, Jon Gonzales, Elena Carano, Chang Liu, Yujia Wu, Fang-Chieh Chou, Tory Smith, Ugo Rosolia, Jacopo Guanetti, Georg Schildbach, and Monimoy Bujarbaruah. Also thanks to my close scientific collaborators Eugen Solowjow, Tammo Zobel, Yutong Li, Shreyas Sudhakar, Akhil Neti, Nikhil Neti, Jeffrey Dinakar, Viktor Rausch, and A. René Geist who greatly influenced my time at Berkeley.

I am grateful for the amazing friendships I get to be a part of in the Bay Area and around the globe. The names of those who supported me throughout my time at Berkeley are too many to list. Special thanks go to Marco Jessen, Antonio Ingram, Lashon Daley, Cynthia Gonzalez, Kara Brodie, Christina Geraci, Joseph Martin, Hanna Towers, Gordon Stack, Sally Carter, Shuk Chan, Tim Chan, Clarrissa Cabansagan, Tony Yammine, David Fernández, Spencer Frank, Abdulrahman Jbaily, Lukas Lebek, Alyssa Novelia, Octavio Narváez-Aroche, Alice Wang, Elkan Hwang, and Sean Turner.

There are no words to describe the thankfulness I feel for my whole family. Special thanks go to my aunts and uncles, my cousins, my grandmother, as well as my three siblings and their wonderful families. I will never forget how my loving and adventurous parents, Regina and Rainer Hansen, supported me throughout the past years at Berkeley – I am incredibly thankful for you two.

Part I

Introduction

Chapter 1

Motivation

The work presented in this dissertation is motivated and outlined in this introductory chapter. First, systems and control research and its application to vehicle dynamics is motivated in order to introduce the reader to the broader background of this dissertation. Second, conceptual foundations and the central idea of developing a novel predictive control scheme with sliding surfaces is discussed. Third, the contributions of this work are explicitly stated and the chapter closes with an overview of the structure of this dissertation.

1.1 Background

Academia and industry are investing tremendous efforts in the development of intelligent vehicle systems for ground, air, and sea vehicles. The current surge in vehicle technology research is motivated by the enormous potential of such systems to benefit society and the high revenue that is expected from the transportation and mobility sector in the future. This includes work on autonomous cars that have the potential to drastically increase roadway safety. Further examples are advanced powertrain research for low and zero emission vehicles as well as novel suspension systems that increase ride comfort. In addition, autonomous or remote controlled robots can be vastly applied for tasks that are dangerous or otherwise undesirable for humans.

The desired progress in the area of vehicle systems requires engineering solutions with more complex features and increased functionality [79]. Hence, the current trends demand for advancements in vehicle software in general and further development and deployment of advanced control techniques in particular. Therefore, vehicle research has become a main incubator for advanced systems and control research at the current time. The importance of systems and control research in light of the current technology innovations was also emphasized in the recent paper [57]. The challenging task of advanced control algorithms is to compute intelligent control actions for possibly complex and nonlinear systems subject to physical and computational budget limits.

1.2 Terminology and Central Idea

The work [57] specifically highlights the role of receding horizon control, also known as model predictive control (MPC), as a promising approach for addressing the current challenges in the automatic control field. The popularity of MPC is due to its ability to account for complicated system dynamics, forecasted environment information, and system constraints. MPC uses the receding horizon technique to approximately solve the infinite horizon optimal control problem. Using a model of the considered system, MPC extrapolates the system evolution over a finite prediction horizon, while accounting for constraints and forecasted environment information. Then, a sequence of future control moves is obtained as the solution to a constrained optimization problem that minimizes a cost function encoding the desired system behavior. Once the optimization problem is solved, part of the resulting solution sequence is applied and by repeating this procedure the plan is revised once new measurements become available. While MPC is extremely powerful, unfortunately, it can be challenging to explicitly prove stability especially for nonlinear and tracking control systems. Another bottleneck is the relatively high computational budget that MPC requires, but this becomes less critical as computing hardware is becoming more and more powerful. While the MPC objective is most commonly formulated as a quadratic function in terms of the states and inputs, this is a somewhat arbitrary choice as the success of MPC is rather linked to the flexibility obtained from its generality [11]. This raises the question whether there are other cost functionals that can prove beneficial in certain control scenarios.

Another method that has attracted the attention of many researchers and engineers especially in the nonlinear control and vehicle dynamics fields is sliding mode control (SMC). The popularity of SMC is inherently linked to its rich theory and its practicality, which allows application to nonlinear and uncertain systems. The essence of SMC lies in formulating a sliding manifold in state space with guaranteed stability. Discontinuous control action is used to steer the system state onto this manifold and render it invariant by confining the state in a sliding motion. In effect, this technique translates a possibly high dimensional tracking control problem to a lower dimensional stabilization control problem [91]. Several extensions to classical SMC were proposed in the past decades that allow for the application of the simple concept of sliding mode control to a variety of systems and control scenarios. A discrete-time counterpart to sliding mode control was established mostly in the 1990s. Different discrete-time sliding control strategies can be found in the literature, see e. g. [28, 29, 2] as well as the comprehensive overview included in [53].

In contrast to sliding mode control, sliding control (SC) uses the concept of a sliding hypersurface for design, but typically no discontinuous control is used. In order to avoid confusion about the terminology, the following quote that is in agreement with standard nonlinear control literature like [91] gives additional insight in the use of the term sliding control versus sliding mode control.

> "We use the term sliding control rather than sliding mode control when the utilized design procedure is generally similar to the sliding mode method. A stable

differential/difference manifold is defined such that on this manifold all trajectories approach the desired trajectory. The control is chosen to drive all trajectories to this manifold. The term sliding mode is used when a discontinuous control is designed that can drive all trajectories to this manifold in finite time forcing the system to enter the sliding mode which is a zero amplitude, infinite frequency motion along the manifold. On the other hand, the term sliding control refers to a smoothed version of the sliding mode control law that drives all trajectories to a boundary layer around the manifold and thus does not necessarily result in perfect asymptotic tracking in the presence of uncertainty."

J. Karl Hedrick

A remaining weakness of sliding control is the difficulty of incorporating constraints on the system's state and input. Setting constraints on the control input and the system's state vector significantly complicates the design of sliding controllers and little research has been done on methods that incorporate constraints. In the previous works [46, 82, 81] the problems of state and output constraints were addressed for sliding control systems, however, the issue of also having a constrained input was not considered. In fact, it is common engineering practice to ignore input constraints during the design phase and simply apply saturation to the computed control signal to ensure that the signal is in some feasible range when implemented. This approach, however, can cause poor system behavior or even instability of the closed-loop. Therefore, finding a systematic way of handling constraints is a highly relevant engineering issue.

Inspired by the complementary characteristics of sliding control and model predictive control, this work merges sliding surfaces into the receding horizon control design process. The resulting control scheme is called receding horizon sliding control (RHSC).

1.3 Main Contributions

This work contributes the development of the receding horizon sliding control technique and studies its application to selected vehicle systems. This dissertation first reviews similar control approaches from the existing literature. Subsequently, expanding on previous works, this dissertation develops RHSC for a wide class of nonlinear systems exploiting invariance properties of sliding surfaces for guaranteeing persistent feasibility and asymptotic stability. Moreover, for linear tracking control problems, the flatness property of sliding surfaces is used to obtain a predictive controller of minimal complexity. The ideas presented in this work are applied to practical control problems. The applications include an autonomous underwater robotics problem, an automotive engine control scenario, and a self-driving vehicle control case study. Simulations and real-world experiments confirm the effectiveness of the developed control methodology. The results indicate that the proposed control scheme is typically easy to tune, behaves well under system uncertainty, and has manageable computational requirements that make it amenable to fast systems with sampling times of the order of fractions of seconds.

This work spans across multiple domains of control theory, specifically those are discrete sliding control (SC), model predictive control (MPC), and existing combinations of the aforementioned two methodologies (SC/MPC). The impact of this work on each individual domain is summarized in the following. The contributions can roughly be categorized in theoretical and application related contributions.

SC: This work contributes a new discrete sliding control approach that is extended with a receding prediction horizon, while most of the standard design and tuning remains unchanged. This way, the popular discrete sliding control approach is extended to constrained nonlinear systems and to systems that require anticipatory control action. Hence, the new approach allows application of discrete sliding control concepts to a far larger class of systems that would normally be hard to control using classical discrete sliding control. Specific examples treated in this dissertation that require anticipatory control action are the path following control problems for an autonomous car and an underwater robot.

MPC: With regard to the model predictive control community, this work contributes a novel predictive control scheme for a wide class of nonlinear systems. In particular, it contributes a suitable choice for the cost function and a systematic way for obtaining invariant sets for nonlinear systems. These two components are the key for proving persistent feasibility and stability but are typically hard to find for general nonlinear systems. Furthermore, for provably stable linear setpoint tracking control, this work reduces the complexity of state-of-the-art methods by reducing the number of necessary optimization variables and constraints. In addition, this work covers an engine control scenario, where RHSC outperforms MPC in terms of tracking performance and average computational effort.

SC/MPC: This work contributes to the theory of combined SC/MPC schemes for nonlinear and MIMO systems, which are rarely covered adequately by the existing approaches. To the author's knowledge, the application of combined SC/MPC approaches to provably stable and persistently feasible constrained tracking control is entirely new. On the application side, this work contributes three new applications that have not been treated with SC/MPC approaches before. Moreover, the experimental validation of a combined SC/MPC method on systems with sampling times of the order of fractions of seconds such as autonomous vehicle control is unprecedented at this point. The observed robust stability in the presence of uncertainty and the good performance in practical applications seen from the case studies in this work support the observations in the SC/MPC literature.

1.4 Outline

The structure of the remainder of this dissertation is detailed in the following. The remaining chapter 2 of this introductory part I gives a review of predictive control fundamentals that this dissertation builds on. Part II of this dissertation develops and analyzes the receding horizon sliding control technique. In particular, chapter 3 contains a literature review on related approaches and receding horizon sliding control is formulated for nonlinear output error regulation control systems. Further theoretical developments for constrained linear tracking control systems are given in chapter 4. Application of receding horizon sliding control to different vehicle systems is discussed in part III. Specifically, chapter 5 discusses the application of RHSC to the three-dimensional path following problem of an autonomous micro-underwater vehicle. In chapter 6, RHSC is applied to a cold-start control problem. Finally, in chapter 7, the RHSC method is applied to an autonomous car for autonomously performing severe maneuvers, both in simulations and real-world experiments. Finally, chapter 8 in part IV summarizes this dissertation and an outlook on possible future research directions is given.

Chapter 2

Predictive Control Fundamentals

This chapter reviews some of the fundamental concepts of model predictive control that are essential for the developments in the remainder of this dissertation. In particular, the chapter contains a discussion of the control methodology's basic algorithm, an introduction to the mathematical notation conventions, and a summary of selected theoretical results.

2.1 Model Predictive Control Algorithm

The conceptual idea of model predictive control, or receding horizon control, is the following. At every time-step, the state of the system of interest is measured. Based on a prediction model of the considered system, the MPC algorithm generates a plan of future actions/controls in an attempt to optimize a given performance index over a finite preview window. The first part of the plan is executed and the system evolves accordingly for one sampling interval. Instead of executing the complete plan of actions, at the next time-step, the algorithm incorporates new information in terms of an updated measurement of the state vector. Starting from the updated state, the MPC algorithm replans the sequence of future controls with the prediction window shifted forward by one step. This procedure repeats at every time-step.

More formally, the standard receding horizon control strategy [9] is summarized in algorithm 1. The main challenge is the design of a constrained finite-time optimal control

Algorithm 1 Online receding horizon control

```
1: repeat
2:     measure the state vector at the current time-step
3:     solve a CFTOC problem and obtain a future control sequence
4:     command the first element of the obtained future control sequence to the plant
5:     let the plant evolve for one sampling interval
6:     increase the current time-step by one
```

(CFTOC) problem [9] that captures the considered control objective within a prediction window. By modifying the CFTOC problem, the concept of predictive control can be applied to different system classes and control scenarios. Selected model predictive control variants are discussed in the following sections of this chapter. First, MPC for regulation problems is addressed. Subsequently, specific MPC formulations for tracking are covered.

2.2 MPC for Regulation Problems

Firstly, consider the most common model predictive control formulation which is concerned with regulation, i. e. the problem of driving the system state to the origin. The cases of nonlinear and linear systems are treated separately in the following.

2.2.1 MPC Regulator for Nonlinear Systems

Consider discrete-time systems with state vector $\boldsymbol{x}(k) \in \mathbb{R}^n$ and input vector $\boldsymbol{u}(k) \in \mathbb{R}^m$, where the time-step k is indicated in parentheses representing the respective signal at time kT_s with T_s denoting the sampling time. The evolution of the system state is governed by equations of the form $\boldsymbol{x}(k) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k))$ that are assumed to have an equilibrium point at the origin, i. e. $\mathbf{0}^{n\times 1} = \boldsymbol{f}(\mathbf{0}^{n\times 1}, \mathbf{0}^{m\times 1})$. Assume the exact prediction model

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.1}$$

is available and state and input constraints are of the form

$$\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n, \tag{2.2}$$

$$\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m. \tag{2.3}$$

Since (2.1) involves model-based predictions of the system's signals rather than the actual signals, a subscript is used to indicate the time-step rather than parentheses.

Using the shorthand notation

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N} \end{bmatrix}, \tag{2.4}$$

$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+N-1} \end{bmatrix}, \tag{2.5}$$

the standard CFTOC problem for MPC regulators can be formulated as

$$\begin{aligned} \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} \quad & p(\boldsymbol{x}_{k+N}) + \sum_{i=0}^{N-1} q(\boldsymbol{x}_{k+i}, \boldsymbol{u}_{k+i}) \\ \text{s. t.} \quad & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i),\ i = k, \dots, k+N-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U},\ i = k, \dots, k+N-1 \\ & \boldsymbol{x}_{k+N} \in \mathcal{X}_f \\ & \boldsymbol{x}_k = \boldsymbol{x}(k). \end{aligned} \tag{2.6}$$

Problem (2.6) minimizes a chosen performance index subject to a set of constraints using the sequences of future states and inputs, $\boldsymbol{X}_k$ and $\boldsymbol{U}_k$, as optimization variables. The cost function in (2.6) is separated in a terminal cost term, $p(\boldsymbol{x}_{k+N})$, and a sum over stage cost terms, $q(\boldsymbol{x}_{k+i}, \boldsymbol{u}_{k+i}), i = 0, \dots N-1$. Furthermore, the constraints in (2.6) include the system dynamics from (2.1) as well as the conditions (2.2), (2.3). Moreover, a terminal constraint involving the terminal set $\mathcal{X}_f \subseteq \mathcal{X}$ is included. The problem is initialized with the current state, $\boldsymbol{x}(k)$, that is assumed to be measurable.

Two key properties for receding horizon controllers are persistent feasibility and asymptotic stability [9]. The following theorems give conditions under which persistent feasibility and asymptotic stability are guaranteed. The theorems rely on invariant set concepts that are defined first.

Definition 2.1 (From [9])**.** A set $\mathcal{O} \subseteq \mathcal{X}$ is said to be a positive invariant set for the autonomous system $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k)$ subject to the constraints $\boldsymbol{x}_k \in \mathcal{X}, \forall k \geq 0$, if

$$\boldsymbol{x}_0 \in \mathcal{O} \quad \Rightarrow \quad \boldsymbol{x}_k \in \mathcal{O}, \forall k \in \mathbb{N}_+. \tag{2.7}$$

Farther, the set $\mathcal{O}_\infty \subseteq \mathcal{X}$ is the maximal positive invariant set for the autonomous system $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k)$ subject to the constraints $\boldsymbol{x}_k \in \mathcal{X}, \forall k \geq 0$, if $\mathcal{O}_\infty$ is invariant and $\mathcal{O}_\infty$ contains all invariant sets contained in $\mathcal{X}$.

Definition 2.2 (From [9])**.** A set $\mathcal{C} \subseteq \mathcal{X}$ is said to be a control invariant set for the system $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k)$ subject to the constraints $\boldsymbol{x}_k \in \mathcal{X}, \boldsymbol{u}_k \in \mathcal{U}, \forall k \geq 0$, if

$$\boldsymbol{x}_0 \in \mathcal{C} \quad \Rightarrow \quad \exists \boldsymbol{u}_k \in \mathcal{U} \,:\, \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \in \mathcal{C}, \forall k \in \mathbb{N}_+. \tag{2.8}$$

Farther, the set $\mathcal{C}_\infty \subseteq \mathcal{X}$ is the maximal control invariant set for the system $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k)$ subject to the constraints $\boldsymbol{x}_k \in \mathcal{X}, \boldsymbol{u}_k \in \mathcal{U}, \forall k \geq 0$, if $\mathcal{C}_\infty$ is control invariant and $\mathcal{C}_\infty$ contains all control invariant sets contained in $\mathcal{X}$.

Theorem 2.1 (From [9])**.** *Consider algorithm 1 with the CFTOC problem* (2.6) *in line 3 and with $N \geq 1$. If $\mathcal{X}_f$ is a control invariant set for system* (2.1), (2.2), (2.3)*, then the receding horizon controller is persistently feasible.*

Proof. The proof is given in [9]. □

Theorem 2.2 (From [9])**.** *Consider algorithm 1 with the CFTOC problem* (2.6) *in line 3 and with $N \geq 1$. Assume that*

- *q and p are positive definite functions;*
- *the sets $\mathcal{X}$, $\mathcal{X}_f$, and $\mathcal{U}$ are closed and contain the origin in their interior;*
- *$\mathcal{X}_f$ is control invariant;*
- *the condition $\min_{\boldsymbol{v} \in \mathcal{U}, \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{v}) \in \mathcal{X}_f} p(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{v})) - p(\boldsymbol{x}) + q(\boldsymbol{x}, \boldsymbol{v}) \leq 0, \forall \boldsymbol{x} \in \mathcal{X}_f$ holds.*

Then, the state of the closed-loop system converges to the origin, i.e. $\lim_{k\to\infty} \boldsymbol{x}(k) = 0$*, and the origin of the closed-loop system is asymptotically stable.*

Proof. The proof is given in [9]. $\square$

Remark 2.1. For a given nonlinear systems, it is often very difficult to compute the terminal invariant set, $\mathcal{X}_f$, and a suitable terminal cost function, $p(\boldsymbol{x})$.

2.2.2 MPC Regulator for Linear Systems

For the case of a linear system with quadratic terminal and stage cost terms as well as polyhedral input and state constraints, the CFTOC problem (2.6) simplifies to the following form,

$$\begin{aligned}\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \; & \|\boldsymbol{x}_{k+N}\|^2_{\boldsymbol{P}} + \sum_{i=0}^{N-1} \|\boldsymbol{x}_{k+i}\|^2_{\boldsymbol{Q}} + \|\boldsymbol{u}_{k+i}\|^2_{\boldsymbol{R}} \qquad (2.9)\\ \text{s. t.} \; & \boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i, \; i = k, \dots, k+N-1\\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U}, \; i = k, \dots, k+N-1\\ & \boldsymbol{x}_{k+N} \in \mathcal{X}_f\\ & \boldsymbol{x}_k = \boldsymbol{x}(k).\end{aligned}$$

The weighted Euclidean norm is used in (2.9) and the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are restricted to be positive definite. The pair $(\boldsymbol{A}, \boldsymbol{B})$ is assumed to be stabilizable.

A standard way to guarantee that the results of theorems 2.1 and 2.2 apply in this simplified case uses the linear quadratic regulator as a terminal control law, i. e. $\boldsymbol{v} = \boldsymbol{K}\boldsymbol{x}$ with $\boldsymbol{K} = -\left(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{B}\right)^{-1} \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{A}$. The matrix $\boldsymbol{P}_\infty$ is found from the algebraic Riccati equation

$$\boldsymbol{P}_\infty = \boldsymbol{A}^\top \boldsymbol{P}_\infty \boldsymbol{A} + \boldsymbol{Q} - \boldsymbol{A}^\top \boldsymbol{P}_\infty \boldsymbol{B}(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{A}. \qquad (2.10)$$

Then, as shown in [9], a suitable terminal cost matrix is $\boldsymbol{P} = \boldsymbol{P}_\infty$. Moreover, denote the maximal positive invariant set of the autonomous system

$$\boldsymbol{x}_{k+1} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})\, \boldsymbol{x}_k \qquad (2.11)$$

subject to the constraints

$$\boldsymbol{x} \in \mathcal{X}, \qquad (2.12)$$

$$\boldsymbol{K}\boldsymbol{x} \in \mathcal{U} \qquad (2.13)$$

as $\mathcal{O}_\infty$. Then, a suitable terminal set is found as $\mathcal{X}_f = \mathcal{O}_\infty$. Hence, for the given simplified scenario with linear system dynamics a standard procedure for finding persistently feasible and asymptotically stable predictive controllers exists.

Remark 2.2. While terminal invariant sets are a powerful concept for ensuring stability and persistent feasibility of receding horizon controllers, they are rarely used in practice. Instead, long prediction horizons are typically used to increase the feasibility region of predictive controllers and simulation trials and experiments are carried out to verify closed-loop stability.

2.3 Tracking MPC for Linear Systems

In a series of publications [66, 67, 23, 24], the authors develop a provably stable and persistently feasible MPC for setpoint tracking of linear systems. The method allows for changing setpoints online, which can lead to infeasibility when using classical MPC regulation schemes. The main result on setpoint tracking MPC from [66, 67, 23, 24] is reviewed in this section.

For the following developments a linear prediction model of the following form is considered,

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k, \tag{2.14}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k. \tag{2.15}$$

The newly added output is denoted $\boldsymbol{y}_k \in \mathbb{R}^p$. The pair $(\boldsymbol{A}, \boldsymbol{B})$ is assumed to be stabilizable. Moreover, polyhedral set constraints of the form

$$\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n, \tag{2.16}$$

$$\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m \tag{2.17}$$

are assumed. Then, for any given target output, $\boldsymbol{r}$, it is ensured that the equation

$$\begin{bmatrix} \boldsymbol{A} - \boldsymbol{I}^{n\times n} & \boldsymbol{B} & \boldsymbol{0}^{n\times p} \\ \boldsymbol{C} & \boldsymbol{D} & -\boldsymbol{I}^{p\times p} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{u}} \\ \boldsymbol{r} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}^{n\times 1} \\ \boldsymbol{0}^{p\times 1} \end{bmatrix} \tag{2.18}$$

has a solution [66]. Solving (2.18) yields the desired setpoint for a given reference value, $\boldsymbol{r}$, and can be described with a parameter vector of minimal size, $\boldsymbol{\theta} \in \mathbb{R}^l$, as

$$\tilde{\boldsymbol{x}} = \boldsymbol{N}_{\boldsymbol{\theta}}\boldsymbol{\theta}, \tag{2.19}$$

$$\tilde{\boldsymbol{u}} = \boldsymbol{M}_{\boldsymbol{\theta}}\boldsymbol{\theta}, \tag{2.20}$$

$$\boldsymbol{r} = \boldsymbol{G}_{\boldsymbol{\theta}}\boldsymbol{\theta}. \tag{2.21}$$

As a next step, the linear quadratic regulator parametrized by $\boldsymbol{\theta}$ is used as a terminal control law, i. e. $\boldsymbol{v} = \boldsymbol{K}(\boldsymbol{x} - \tilde{\boldsymbol{x}}) + \tilde{\boldsymbol{u}} = \boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\boldsymbol{\theta}$ with $\boldsymbol{L} = -\boldsymbol{K}\boldsymbol{N}_{\boldsymbol{\theta}} + \boldsymbol{M}_{\boldsymbol{\theta}}$. For given positive definite matrices $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$, the algebraic Riccati equation,

$$\boldsymbol{P}_\infty = \boldsymbol{A}^\top \boldsymbol{P}_\infty \boldsymbol{A} + \boldsymbol{Q} - \boldsymbol{A}^\top \boldsymbol{P}_\infty \boldsymbol{B}(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{A}, \tag{2.22}$$

yields $\boldsymbol{P}_\infty$. Set $\boldsymbol{P} = \boldsymbol{P}_\infty$ and $\boldsymbol{K} = -(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{P}_\infty \boldsymbol{A}$ and define an augmented state vector as $\boldsymbol{w} := \begin{bmatrix} \boldsymbol{x}^\top & \boldsymbol{\theta}^\top \end{bmatrix}^\top$. Then, the dynamics of the augmented system become

$$\boldsymbol{w}_{k+1} = \begin{bmatrix} \boldsymbol{A} + \boldsymbol{B}\boldsymbol{K} & \boldsymbol{B}\boldsymbol{L} \\ \boldsymbol{0}^{l \times n} & \boldsymbol{I}^{l \times l} \end{bmatrix} \boldsymbol{w}_k =: \boldsymbol{A}^{\text{aug}} \boldsymbol{w}_k \tag{2.23}$$

subject to the constraint

$$\boldsymbol{w} \in \mathcal{W}, \tag{2.24}$$

where

$$\mathcal{W} := \left\{ \boldsymbol{w} \,:\, \begin{bmatrix} \boldsymbol{I}^{n \times n} & \boldsymbol{0}^{n \times l} \end{bmatrix} \boldsymbol{w} \in \mathcal{X}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{L} \end{bmatrix} \boldsymbol{w} \in \mathcal{U} \right\}. \tag{2.25}$$

The so-called invariant set for tracking is then found as the maximal positive invariant set of (2.23) subject to (2.24) as

$$\mathcal{T} := \left\{ \boldsymbol{w} \,:\, \boldsymbol{A}^{\text{aug}k} \boldsymbol{w} \in \mathcal{W}, \forall k \geq 0 \right\}. \tag{2.26}$$

Then, the fundamental idea from [66] is to introduce an additional optimization variable, $\tilde{\boldsymbol{\theta}}$, and to allow for reference offset that is penalized with an additional term in the cost function. This way, persistent feasibility can be maintained even during large reference jumps. The corresponding CFTOC problem for the tracking MPC can be formulated as follows [24],

$$\begin{aligned} \min_{\boldsymbol{X}_k, \boldsymbol{U}_k, \tilde{\boldsymbol{\theta}}} \quad & \|\boldsymbol{x}_{k+N} - \boldsymbol{N}_{\boldsymbol{\theta}} \tilde{\boldsymbol{\theta}}\|_{\boldsymbol{P}}^2 + \sum_{i=0}^{N-1} \|\boldsymbol{x}_{k+i} - \boldsymbol{N}_{\boldsymbol{\theta}} \tilde{\boldsymbol{\theta}}\|_{\boldsymbol{Q}}^2 + \|\boldsymbol{u}_{k+i} - \boldsymbol{M}_{\boldsymbol{\theta}} \tilde{\boldsymbol{\theta}}\|_{\boldsymbol{R}}^2 + \|\boldsymbol{T}(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta})\|_\infty \\ \text{s. t.} \quad & \boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i, \ i = k, \ldots, k+N-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U}, \ i = k, \ldots, k+N-1 \\ & \begin{bmatrix} \boldsymbol{x}_{k+N}^\top & \tilde{\boldsymbol{\theta}}^\top \end{bmatrix}^\top \in \mathcal{T} \\ & \boldsymbol{x}_k = \boldsymbol{x}(k), \end{aligned} \tag{2.27}$$

where $\boldsymbol{\theta}$ is such that $\boldsymbol{r} = \boldsymbol{G}_{\boldsymbol{\theta}} \boldsymbol{\theta}$. The matrix $\boldsymbol{T} \in \mathbb{R}^{l \times l}$ involved in the offset cost function is restricted to be designed as non-singular. Using (2.27) as part of a receding horizon control scheme yields a persistently feasible controller with desirable convergence properties as stated in the following theorem.

Theorem 2.3 (From [24]). *Consider algorithm 1 with the CFTOC problem* (2.27) *in line 3 and with $N \geq 1$. If the given target operating point, $\boldsymbol{r}$, is such that $\boldsymbol{\theta} \in \mathcal{O}$, then for any feasible initial state the closed-loop system asymptotically evolves to the target operating point. If the given target operating point, $\boldsymbol{r}$, is such that $\boldsymbol{\theta} \notin \mathcal{O}$, then for any feasible initial state the closed-loop system asymptotically evolves to the operating point $\tilde{\boldsymbol{\theta}}^*$ that minimizes the offset cost function, i. e.*

$$\tilde{\boldsymbol{\theta}}^* = \arg \min_{\tilde{\boldsymbol{\theta}} \in \mathcal{O}} \|\boldsymbol{T}(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta})\|_\infty. \tag{2.28}$$

The set $\mathcal{O}$ is given by $\mathcal{O} := \left\{ \boldsymbol{\theta} \,:\, \begin{bmatrix} \boldsymbol{N}_{\boldsymbol{\theta}} \\ \boldsymbol{I}^{l \times l} \end{bmatrix} \boldsymbol{\theta} \in \mathcal{T} \right\}$.

Proof. The proof is given in [24]. □

2.4 Delta Input Formulation of MPC

For general nonlinear output tracking problems it is often difficult to obtain a state and input reference trajectory. A popular alternative MPC scheme in this case is the so-called delta input formulation [9]. Assume a state space prediction model of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{2.29}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.30}$$

with the constraints

$$\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n, \tag{2.31}$$

$$\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m. \tag{2.32}$$

The idea of the delta input formulation is to penalize the deviation of the system output from its desired value and the change in the input signal. Using e. g. the weighted Euclidean norm, the resulting scheme reads

$$\begin{aligned}
\min_{\boldsymbol{X}_k, \boldsymbol{U}_k} \quad & \|\boldsymbol{y}_{k+N} - \boldsymbol{r}_{k+N}\|_Q^2 + \sum_{i=0}^{N-1} \|\boldsymbol{y}_{k+i} - \boldsymbol{r}_{k+i}\|_Q^2 + \|\boldsymbol{u}_{k+i} - \boldsymbol{u}_{k+i-1}\|_R^2 \\
\text{s. t.} \quad & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i),\ i = k, \dots, k+N-1 \\
& \boldsymbol{y}_i = \boldsymbol{h}(\boldsymbol{x}_i, \boldsymbol{u}_i),\ i = k, \dots, k+N \\
& \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U},\ i = k, \dots, k+N-1 \\
& \boldsymbol{x}_{k+N} \in \mathcal{X}_f \\
& \boldsymbol{x}_k = \boldsymbol{x}(k) \\
& \boldsymbol{u}_{k-1} = \boldsymbol{u}(k-1).
\end{aligned} \tag{2.33}$$

Again, scheme (2.33) includes the system's state transition model as a constraint. Due to the inclusion of the output in the cost function, the output model is added as a constraint as well. In addition to the standard state, input, terminal, and initial constraints, the past control has to be initialized in order to evaluate the cost function in (2.33). As before, an MPC can be obtained by using the CFTOC problem (2.33) in line 3 of algorithm 1.

Part II

Receding Horizon Sliding Control Theory

Chapter 3

RHSC for Nonlinear Systems

This chapter formalizes receding horizon sliding control for output error regulation of nonlinear systems. In particular, the concept of sliding hypersurfaces is used to translate this output tracking goal into a state space objective. Sliding hypersurfaces are designed to encode desired tracking error dynamics, hence, the design and tuning is easy and intuitive. By applying a predictive control framework, the controller minimizes the deviation of the system's state to the designed reference surfaces. Constraint satisfaction is directly inherited from nonlinear MPC. Furthermore, stability and persistent feasibility are proven by exploiting the invariance property of sliding hypersurfaces. A simple motion control problem is used to illustrate the effectiveness of the proposed control methodology. Parts of this chapter have been previously published in the works [39] and [94].

3.1 Motivation and Literature Review

The idea of combining sliding control with so-called soft computing methods such as neural networks, fuzzy logic or probabilistic reasoning, has resulted in broad research efforts and enormous output in terms of publications as the survey paper [100] reports. Along a similar line of thought but far less exposed in the literature stands the idea of combining sliding control with core concepts from model predictive control. Hybrid control methodologies proposed so far can be roughly categorized in two groups. The first group utilizes a receding horizon framework together with a sliding surface cost function. The second class of algorithms have a model predictive controller at its core and use sliding control in an outer loop that provides robustness to the overall control scheme. This work follows the first research path. Existing works along this line of thought are reviewed next. The interested reader is referred to [84, 65, 64] for further details on the second research path, which is mostly concerned with mitigating the computational burden and conservativeness of robust MPC.

The fact that the characteristics of MPC and SMC are complementary and the idea of combining both methods in one control scheme was pointed out in [102]. In this paper, a control method called sliding mode model predictive control is proposed for linear multi-input systems. The controller uses a constrained optimization problem with a performance index that penalizes the distance of the system state from a sliding surface as well as the control effort to reach this surface. Once the state is on the sliding surface, it remains there and slides to the origin. The paper [101] extends the ideas from [102] to nonlinear multi-input systems. While pioneering the idea of combining model predictive control and sliding control, the works [102, 101] are incomplete, e. g. as pointed out in [98] the persistent feasibility property is not considered. Persistent or recursive feasibility ensures that feasibility of the underlying optimization problem at the initial time-step yields feasibility at all successive time-steps. Since this condition is also essential for stability, the stability proofs in [102, 101] are not adequate. In [98] a thorough treatment of the persistent feasibility and stability properties is given for linear single-input systems. Moreover, it is shown that the robustness properties of SMC can be exploited in a receding horizon control formulation by using a min-max formulation of the corresponding constrained optimization problem. The resulting methodology is named model predictive sliding mode control and as in [102, 101] the functionality of the methodology is demonstrated with a simple simulation example.

Further related work has appeared in [43], where an explicit discrete sliding controller with prediction horizon was derived but it only applies to unconstrained linear systems. On the other hand, in [16] generalized predictive control is combined with variable structure control for linear single-input single-output (SISO) systems. Successful simulations of the resulting controller applied to a chemical process justify the novel design. It is reported that the resulting controller inherits anticipatory actuation from generalized predictive control and has good robustness under parameter variations. The paper [78] develops a so-called predictive sliding mode controller for first-order plus dead time systems that combines generalized predictive control with discrete sliding mode control. It is emphasized that the resulting controller only requires a small number of tuning parameters. In [76], predictive sliding mode controllers are applied using a SISO first-order plus dead time model of a solar energy plant. The resulting control strategy is successfully tested on a distributed solar collector field. In the paper [31], a method called sliding mode predictive control for linear SISO systems is applied to a solar air conditioning plant. The authors report that the resulting controller has good robustness properties and successfully handles setpoint changes, without requiring bigger computational resources than classical MPC. The experimental tests reported in [76] and [31] both utilize sampling intervals of the order of several seconds.

This chapter builds on the existing literature [102, 101, 98] and uses sliding surfaces for deriving predictive controllers. However, the resulting design from this work is applicable to a wide class of multi-input multi-output (MIMO) nonlinear systems. In essence, sliding surfaces are used to generate a reference location in state space based on the desired output signal. The development leads to a receding horizon cost function, which penalizes the distance of the system state from this reference manifold. On the surfaces, predefined error dynamics hold that ensure tracking error decrease as desired. A detailed analysis shows that the developed cost function is a suitable Lyapunov-like function for analyzing stability. Moreover, a subset of the sliding surface that remains invariant in the presence of system constraints is extracted from the sliding surface and utilized as a terminal set within a predictive control framework. The modified cost and the novel terminal set yield the key components for deriving a provably stable and persistently feasible predictive control scheme. The developed control method is referred to as receding horizon sliding control.

The remainder of this chapter is structured as follows. Section 3.2 describes the concept of sliding surfaces for nonlinear SISO and MIMO systems. RHSC for SISO and MIMO regulation problems is discussed in sections 3.3 and 3.4, respectively. These sections also include persistent feasibility and stability proofs. Subsequently, the proposed control algorithm is further illustrated in section 3.5 with a simple nonlinear motion control example. Concluding remarks on this chapter are given in section 3.6.

3.2 Sliding Surfaces for Nonlinear Systems

This section introduces sliding surfaces for nonlinear systems, which is a key concept in the development of RHSC in this chapter. Based on nonlinear system equations, sliding variables and sliding hypersurfaces are derived. The section first covers single-input single-output systems and, subsequently, the presented ideas are extended to the multi-input multi-output case.

3.2.1 SISO Systems

Consider discrete-time SISO dynamical systems in state space form. The constrained state vector is denoted by $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n$, the constrained input signal is $u \in \mathcal{U} \subseteq \mathbb{R}$, and $y \in \mathbb{R}$ denotes the output. The sampling time is T_s and the system's signals at time kT_s are represented by $\boldsymbol{x}(k)$, $u(k)$, and $y(k)$. For simplicity it is assumed that $\boldsymbol{x}(k)$ and $y(k)$ are exactly known at the current time-step.

Furthermore, let

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, u_k), \tag{3.1}$$

$$y_k = h(\boldsymbol{x}_k) \tag{3.2}$$

constitute an exact prediction model for extrapolating the system evolution into the future. Subscripts rather than brackets indicate the time-steps in (3.1), (3.2) in order to emphasize that these equations are a prediction model for the system's signals. Moreover, the smooth state transition map and output function are symbolized by $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ and $h : \mathbb{R}^n \to \mathbb{R}$, respectively. The following assumptions are imposed on the system equations (3.1), (3.2).

Assumption 3.1. Everywhere in the feasible set $\mathcal{X}$, the system (3.1), (3.2)

- has well-defined and fixed relative degree d;
- is globally minimum phase.

The relative degree for discrete nonlinear SISO systems is the number of delay steps that it takes for the input to directly affect the output, see [49] for the formal definition. The minimum phase assumption ensures that the system's zero dynamics are asymptotically stable.

The control goal consists of letting the system track a desired output reference. Different from the classical predictive control literature the concept of sliding hypersurfaces is used for approaching the tracking objective in this work. In particular, the surface design procedure from [89, 90] is adopted in this work. Conceptually, a sliding surface is defined as a manifold in state space, where stable output error dynamics hold.

Let the tracking error be denoted by

$$e_k = y_k - r_k, \tag{3.3}$$

representing the difference between the system output from (3.2) and its predefined desired reference value, r_k. Using the shorthand notation $\mathscr{D}(e)_k$, let a general output error difference equation of order $l = d - 1$ be defined as follows [54],

$$\mathscr{D}(e)_k := e_{k+l} - \delta(e_k, \dots, e_{k+l-1}) = 0. \tag{3.4}$$

The function $\delta : \mathbb{R}^l \to \mathbb{R}$ is to be chosen by the designer to encode desired error dynamics for the system. It is required that the design is such that $e_k = 0$ is a globally asymptotically stable equilibrium point of $e_{k+l} = \delta(e_k, \dots, e_{k+l-1})$ [54]. Hence, in addition to stability in the sense of Lyapunov it can be concluded that

$$\mathscr{D}(e)_k = 0, \forall k \quad \Rightarrow \quad \lim_{k\to\infty} e_k = 0. \tag{3.5}$$

Then, a sliding variable is readily defined as

$$s_k = \mathscr{D}(e)_k. \tag{3.6}$$

An alternative representation of the sliding variable can be obtained through recursive substitution using the relations (3.1), (3.2), and (3.3). Then the sliding variable (3.6) can be rewritten as a function of the state vector at time-step k and the desired output,

$$s_k = \Psi(\boldsymbol{x}_k, r_k, \dots, r_{k+l}). \tag{3.7}$$

The chosen design procedure guarantees that s_k from (3.7) has relative degree equal to one. In other words, $s_{k+1} = \Psi(\boldsymbol{x}_{k+1}, r_{k+1}, \dots, r_{k+l+1})$ is an explicit function of the control u_k after substituting (3.1).

As in [91], driving s_k to zero simultaneously corresponds to enforcing the predefined error dynamics $\mathscr{D}(e)_k = 0$ and to forcing the system to approach a hypersurface in state space that is parametrized by the reference signal and given by $\Psi(\boldsymbol{x}_k, r_k, \dots, r_{k+l}) = 0$. Figure 3.1 illustrates a two-dimensional sliding surface in a three-dimensional state space. As the depicted exemplary trajectory indicates, once the state is on the sliding surface, the designed error dynamics apply and the system slides to the desired setpoint.

Next, a similar procedure is discussed for MIMO systems.

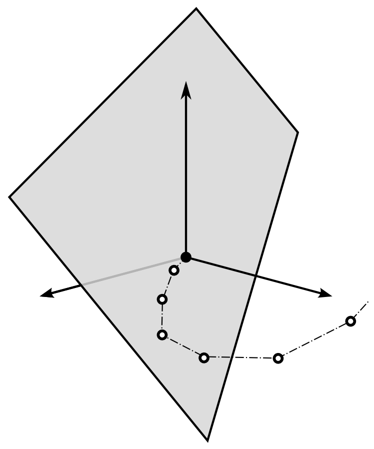

Figure 3.1: Schematic of a two dimensional sliding surface in a three dimensional state space as well as an exemplary trajectory of the state approaching the surface and sliding to the origin once it is on the surface

3.2.2 MIMO Systems

Consider the class of prediction models that can be defined by the nonlinear MIMO discrete-time system equations

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{3.8}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k). \tag{3.9}$$

The state vector is again denoted by $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n$, the input vector is $\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m$, and the output vector is $\boldsymbol{y} \in \mathbb{R}^p$, where $\mathcal{X}$ and $\mathcal{U}$ contain the feasible state and control vectors as usual. The state transition function is now denoted as $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and the measurement function is defined as $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^p$. As before, the discrete time-step is indicated by the subscripts k and $k+1$, respectively. Additionally, the following assumption is made.

Assumption 3.2. Everywhere in the feasible set $\mathcal{X}$, the system (3.8), (3.9)

- has well-defined and fixed relative degree $(d_1, \ldots, d_p)$;
- is globally minimum phase.

The control goal consists of steering the system to desired output trajectories. Formally, in the MIMO case this corresponds to driving the vector

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \boldsymbol{r}_k \tag{3.10}$$

to zero. The signal $\boldsymbol{e} \in \mathbb{R}^p$ consists of the p tracking errors between outputs, $\boldsymbol{y}_k$, and desired reference outputs, $\boldsymbol{r}_k$.

Extending the design method from section 3.2.1 to MIMO systems, desired dynamics are selected for each output error. In particular, let $\delta_i(e_{i,k}, \ldots, e_{i,k+l_i-1})$, $i = 1, \ldots, p$ denote p functions of the tracking error components analogous to the previous section. The l_i equal the relative degree d_i of the i^{th} output minus one, i. e. $l_i = d_i - 1$. Using the shorthand operator notation $\mathscr{D}(\boldsymbol{e})_k$, the above is formalized as follows,

$$\mathscr{D}(\boldsymbol{e})_k := \begin{bmatrix} e_{k+l_1} - \delta_1(e_{1,k}, \ldots, e_{1,k+l_1-1}) \\ \vdots \\ e_{k+l_p} - \delta_p(e_{p,k}, \ldots, e_{p,k+l_p-1}) \end{bmatrix}. \tag{3.11}$$

The functions in (3.11) are to be chosen by the designer. Thereby, it is required that the p difference equations in the tracking errors, $\mathscr{D}(\boldsymbol{e})_k = 0$, are globally asymptotically stable. In addition to stability, it is required that the functions δ_i are chosen such that they are of relative degree one, which is analogous to the SISO case.

The sliding variable vector is now readily defined as

$$\boldsymbol{s}_k = \mathscr{D}(\boldsymbol{e})_k. \tag{3.12}$$

Due to the relative degree requirement it is ensured that all components of $\boldsymbol{s}_{k+1}$ are explicit functions of at least one control input.

Similar to the previous section, through recursive substitution the sliding variable vector can be written solely in terms of the state vector at time-step k and the reference as

$$\boldsymbol{s}_k = \boldsymbol{\Psi}(\boldsymbol{x}_k, \boldsymbol{r}_k, \ldots, \boldsymbol{r}_{k+l}). \tag{3.13}$$

Note that the set of all points $\boldsymbol{x}_k$ that satisfy $\boldsymbol{\Psi}(\boldsymbol{x}_k, \boldsymbol{r}_k, \ldots, \boldsymbol{r}_{k+l}) = \boldsymbol{0}^{p\times 1}$ corresponds to the intersection of p hypersurfaces in state space, where $l = \max_{i\in 1,\ldots,p} l_i$. The control goal simply consists of driving $\boldsymbol{s}_k$ to this intersection and exploiting the fact that stable output error dynamics result on this manifold. Figure 3.2 illustrates this for the example of the one-dimensional intersection of two two-dimensional sliding surfaces in a three-dimensional state space. As the depicted exemplary trajectory indicates, once the state is on the intersection of sliding surfaces, the designed error dynamics apply and the system slides to the desired setpoint.

3.3 RHSC for SISO Regulation Problems

In the following, the receding horizon sliding control strategy is formalized for SISO regulation problems. First, invariant sets related to sliding surfaces are introduced for the considered constrained setting. Subsequently, the RHSC scheme is formulated. Finally, persistent feasibility and stability of the proposed control strategy are analyzed.

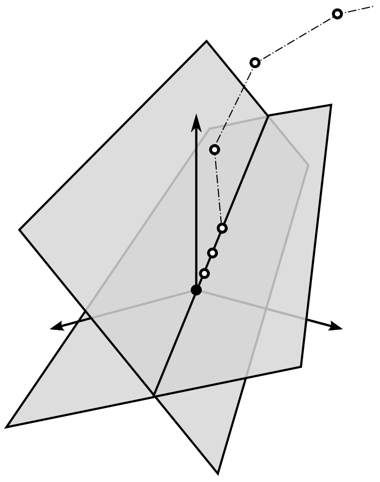

Figure 3.2: Schematic of the one-dimensional intersection of two two-dimensional sliding surfaces in a three-dimensional state space as well as an exemplary trajectory of the state approaching the intersection and sliding to the origin once it is on the intersection

3.3.1 Terminal Invariant Set

In the unconstrained case, it is well known that sliding control drives the system to invariant subsets of the state space [91]. In the following, this idea is formalized in a constrained setting, where state constraints as well as limited control authority are taken into account. Under these conditions only a subset of the sliding hypersurface remains invariant as illustrated in figure 3.3.

The so-called equivalent control is the input that renders the sliding hypersurface invariant. In other words, the equivalent control, u_k^{eq}, is defined such that it solves the scalar algebraic equation $s_{k+1} = 0$, that means it satisfies

$$\Psi(\boldsymbol{f}(\boldsymbol{x}_k, u_k^{\mathrm{eq}}), r) = 0 \tag{3.14}$$

Note that it was used that SISO regulation problems are characterized by $r_k = r$ being constant, i. e. the quantity from (3.7) simplifies according to $\Psi(\boldsymbol{x}_k, r_k, \ldots, r_{k+l}) = \Psi(\boldsymbol{x}_k, r)$.

In general, existence of u_k^{eq} is not guaranteed. For a given r the equivalent control depends on $\boldsymbol{x}_k$ and the dynamics of the closed-loop system under the equivalent control are defined as

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, u_k^{\mathrm{eq}}) =: \boldsymbol{f}^{\mathrm{eq}}(\boldsymbol{x}_k), \tag{3.15}$$

which is only valid in the set

$$\mathcal{X}^{\mathrm{eq}} := \{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{X}, \exists u^{\mathrm{eq}} \in \mathcal{U} \text{ such that } \Psi(\boldsymbol{f}(\boldsymbol{x}, u^{\mathrm{eq}}), r) = 0\}\,. \tag{3.16}$$

In the set $\mathcal{X}^{\mathrm{eq}}$ it is ensured that the state constraints are satisfied and that the equivalent control exists [96].

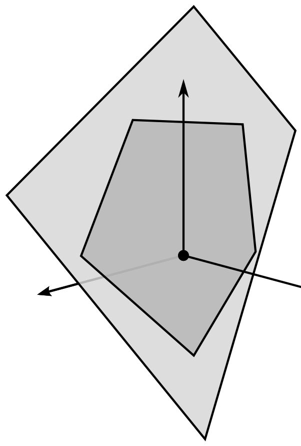
Figure 3.3: Schematic of the invariant subset (dark gray) of a two-dimensional sliding surface (light gray) in a three dimensional state space

Next, define the maximal positive invariant set of system (3.15) subject to (3.16) as

$$\mathcal{S} := \left\{ \boldsymbol{x} : \boldsymbol{f}^{\mathrm{eq}k}(\boldsymbol{x}) \in \mathcal{X}^{\mathrm{eq}}, \forall k \geq 0 \right\}, \tag{3.17}$$

where the notion of an iterated function was used, i. e. $\boldsymbol{g}^q(\boldsymbol{x}) = \left(\boldsymbol{g} \circ \boldsymbol{g}^{q-1}\right)(\boldsymbol{x})$ and $(\boldsymbol{g} \circ \boldsymbol{g})(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{g}(\boldsymbol{x}))$. Note that $\mathcal{S}$ is a fully dimensional set, it contains the invariant part of the sliding hyperplane,

$$\bar{\mathcal{S}} := \left\{ \boldsymbol{x} : \Psi(\boldsymbol{x}, r) = 0, \boldsymbol{f}^{\mathrm{eq}k}(\boldsymbol{x}) \in \mathcal{X}^{\mathrm{eq}}, \forall k \geq 0 \right\}, \tag{3.18}$$

and all states that can be steered to $\bar{\mathcal{S}}$ in one step. In other words, $\mathcal{S}$ is the one-step controllable set [9] of the system (3.15) subject to (3.16) for the target set $\bar{\mathcal{S}}$, i. e. $\mathcal{S} = \mathcal{K}_1(\bar{\mathcal{S}})$.

3.3.2 Controller Formulations

Define $\boldsymbol{S}_{k+1}$ as the matrix containing the quantity $s_{k+i}, i = 1, \dots, N$ over an N-step prediction horizon,

$$\boldsymbol{S}_{k+1} := \begin{bmatrix} s_{k+1} & \dots & s_{k+N} \end{bmatrix}. \tag{3.19}$$

In principle, either one of the equivalent definitions (3.6) or (3.7) can be used in (3.19) leading to slightly different controller formulations.

First, consider the case where (3.7) is used as the sliding variable definition in the future sequence (3.19) and let

$$\boldsymbol{X}_k := \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N} \end{bmatrix}, \tag{3.20}$$

$$\boldsymbol{U}_k := \begin{bmatrix} u_k & \dots & u_{k+N-1} \end{bmatrix}. \tag{3.21}$$

A CFTOC problem for the regulation case can be formulated as

$$\begin{aligned}
\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \quad & \|\boldsymbol{S}_{k+1}^\top\|_2^2 \\
\text{s. t.} \quad & s_i = \Psi(\boldsymbol{x}_i, r),\ i = k+1,\dots,k+N \\
& \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, u_i),\ i = k,\dots,k+N-1 \\
& \boldsymbol{x}_i \in \mathcal{X}, u_i \in \mathcal{U},\ i = k,\dots,k+N-1 \\
& \boldsymbol{x}_{k+N} \in \mathcal{S} \\
& \boldsymbol{x}_k = \boldsymbol{x}(k).
\end{aligned} \tag{3.22}$$

Due to recursive substitutions involved in obtaining (3.7) from (3.6), it is often more compact to use (3.6) in (3.19) for nonlinear systems. Hence, this approach will be further discussed as well. Therefore, let $\bar{N} = N + l$ and rewrite the sequence of future state vectors and control signals as

$$\boldsymbol{X}_k := \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+\bar{N}} \end{bmatrix}, \tag{3.23}$$

$$\boldsymbol{U}_k := \begin{bmatrix} u_k & \dots & u_{k+\bar{N}-1} \end{bmatrix}. \tag{3.24}$$

Note that it is necessary to introduce a small number of additional optimization variables at the end of the prediction horizon in this case compared to (3.20), (3.21). A CFTOC problem equivalent to (3.22) for this case can be formulated as

$$\begin{aligned}
\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \quad & \|\boldsymbol{S}_{k+1}^\top\|_2^2 \\
\text{s. t.} \quad & s_i = \mathscr{D}(e)_i,\ i = k+1,\dots,k+N \\
& e_i = h(\boldsymbol{x}_i) - r,\ i = k+1,\dots,k+\bar{N} \\
& \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, u_i),\ i = k,\dots,k+\bar{N}-1 \\
& \boldsymbol{x}_i \in \mathcal{X}, u_i \in \mathcal{U},\ i = k,\dots,k+\bar{N}-1 \\
& \boldsymbol{x}_{k+N} \in \mathcal{S} \\
& \boldsymbol{x}_k = \boldsymbol{x}(k).
\end{aligned} \tag{3.25}$$

The cost function in (3.22) and (3.25) is chosen to be a measure of the system's deviation from the sliding surface over the prediction horizon. As in classical MPC, at every discrete time-instant, k, the RHSC obtains a sequence of planned future control inputs and states, $\boldsymbol{U}_k^*$ and $\boldsymbol{X}_k^*$, from solving (3.22) or (3.25). The minimization problems (3.22) and (3.25) are dependent on the current state, $\boldsymbol{x}(k)$. The control applied to the system is set as the first component of the obtained future control sequence, i. e. $u(k) = u_k^*$. At the next time-step the same problem is solved with the prediction horizon shifted by one step, compare algorithm 2. This control approach is referred to as receding horizon sliding control [39].

Remark 3.1. Equations (3.22) and (3.25) consider the RHSC objective in terms of the L^2 norm. However, the ideas presented here and in the remainder of this dissertation are not limited to the L^2 case but other norms such as the L^1 or L^∞ norm can be used as well. Also, a weighted norm can be utilized if desired.

Remark 3.2. Comparing scheme (3.22) and (2.6) it can be seen that RHSC can be cast in the form of a model predictive control scheme. However, the fact that $s_k = 0$ is a difference equation in terms of the tracking error introduces terms that correlate errors at different time-steps. Such terms are typically hard to design in classical model predictive control but they are inherently linked to the system response speed and stability. Therefore, the above scheme can be thought of as a systematic way to design these cross correlated terms.

Algorithm 2 On-line receding horizon sliding control for nonlinear SISO regulation problems

```
1: repeat
2:     measure $\boldsymbol{x}(k)$ at time-step $k$
3:     solve (3.22) (or (3.25)) and obtain $\boldsymbol{U}_k^*$
4:     extract first element $u_k^*$ of $\boldsymbol{U}_k^*$ and set $u(k) = u_k^*$
5:     let the system evolve to $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), u(k))$
6:     increase the time-step such that $k \leftarrow k+1$
```

3.3.3 Analysis

While the CFTOC problems (3.22) and (3.25) are equivalent, the analysis will be carried out for scheme (3.22) in the following. The theorems below guarantee persistent feasibility and asymptotic stability of the proposed regulator scheme. The proofs follow a similar procedure as used for classical nonlinear MPC schemes [12], but the invariance and stability properties of the sliding hypersurfaces are exploited here. Moreover, it is shown that the use of sliding surfaces in the receding horizon control design yields an apparent Lyapunov-like function that can be used for guaranteeing stability.

Theorem 3.1. *Given that the initial state, $\boldsymbol{x}_0 = \boldsymbol{x}(0)$, is feasible, the RHSC regulator given in algorithm 2 is persistently feasible.*

Proof. The proof follows from the invariance of the set $\mathcal{S}$. While it may be obvious to some readers, the proof will be carried out in detail for completeness of the exposition.

Feasibility at the initial time-step implies the existence of optimal state and input sequences, $\boldsymbol{X}_0^* = \begin{bmatrix}\boldsymbol{x}_0^* & \dots & \boldsymbol{x}_N^*\end{bmatrix}$ and $\boldsymbol{U}_0^* = \begin{bmatrix}u_0^* & \dots & u_{N-1}^*\end{bmatrix}$, such that all constraints are satisfied, i. e.

$$\boldsymbol{x}_i^* \in \mathcal{X},\ i = 0, \dots, N-1, \quad \boldsymbol{x}_N^* \in \mathcal{S}, \tag{3.26}$$

$$u_i^* \in \mathcal{U},\ i = 0, \dots, N-1. \tag{3.27}$$

Executing $u(0) = u_0^*$ lets the system evolve to $\boldsymbol{x}(1) = \boldsymbol{x}_1^*$, since the dynamics model (3.1) is assumed to be exact.

At $k = 1$, the optimal control sequence $\boldsymbol{U}_1^*$ is sought. A candidate sequence is

$$\boldsymbol{U}_1^\circ = \begin{bmatrix}u_1^* & \dots & u_{N-1}^* & u_N^{\mathrm{eq}}\end{bmatrix}. \tag{3.28}$$

From (3.27), the first $N-1$ entries in $\boldsymbol{U}_1^\circ$ satisfy the input constraints. Furthermore, since $\boldsymbol{x}_N^* \in \mathcal{S} \subseteq \mathcal{X}^{\mathrm{eq}}$, it follows from the definition (3.16) that the equivalent control exists and satisfies the constraint $u_N^{\mathrm{eq}} \in \mathcal{U}$. Likewise, the resulting sequence of states, $\boldsymbol{X}_1^\circ = \begin{bmatrix}\boldsymbol{x}_1^* & \dots & \boldsymbol{x}_N^* & \boldsymbol{x}_{N+1}^\circ\end{bmatrix}$, satisfies the state constraints. This was already verified in (3.26) except for the last entry in $\boldsymbol{X}_1^\circ$, which is $\boldsymbol{x}_{N+1}^\circ$. But since $\mathcal{S}$ is invariant, definition (3.16) can be used to conclude that $\boldsymbol{x}_{N+1}^\circ \in \mathcal{S}$ given that $\boldsymbol{x}_N^* \in \mathcal{S}$ and the equivalent control u_N^{eq} is applied.

In summary, feasibility at the initial time-step implies feasibility at the subsequent time-step. The above procedure can be applied recursively and therefore existence of feasible control and state sequences is guaranteed for all future times. □

Theorem 3.2. *The output of the closed-loop system converges to its desired value, i. e.* $\lim_{k\to\infty} y_k = r$*, and the closed-loop system is asymptotically stable.*

Proof. To prove asymptotic stability of the RHSC regulator scheme, a Lyapunov-like [6] analysis is utilized. Define a scalar function

$$V(\boldsymbol{x}(k), \boldsymbol{U}_k) = \|\boldsymbol{S}_{k+1}^\top\|_2^2 \tag{3.29}$$

with optimal value $V^*(\boldsymbol{x}(k))$, where the optimizers are denoted by $\boldsymbol{X}_k^* = \begin{bmatrix}\boldsymbol{x}_k^* & \dots & \boldsymbol{x}_{k+N}^*\end{bmatrix}$ and $\boldsymbol{U}_k^* = \begin{bmatrix}u_k^* & \dots & u_{k+N-1}^*\end{bmatrix}$.

At the next time-step, $k+1$, a feasible but possibly suboptimal control sequence is $\boldsymbol{U}_{k+1}^\circ = \begin{bmatrix}u_{k+1}^* & \dots & u_{k+N-1}^* & u_{k+N}^{\mathrm{eq}}\end{bmatrix}$ and it follows that

$$V^*(\boldsymbol{x}(k+1)) \le V(\boldsymbol{x}(k+1), \boldsymbol{U}_{k+1}^\circ) = V^*(\boldsymbol{x}(k)) - s_{k+1}^2. \tag{3.30}$$

In (3.30), the fact that $s_{k+N+1} = 0$ was used and hence this term does not feature in (3.30), because the equivalent control, u_{k+N}^{eq}, is applied as part of the sequence $\boldsymbol{U}_{k+1}^\circ$. Note that V^* is decreasing unless $s_{k+1} = 0$. But $s_{k+1} = 0$ yields that the system state is such that $\boldsymbol{x}(k) \in \mathcal{X}^{\mathrm{eq}}$. If $\boldsymbol{x}_k \in \mathcal{X}^{\mathrm{eq}} \cup \mathcal{S}$, it can be immediately concluded that $\|\boldsymbol{S}_{i+1}^\top\|_2^2 = 0, \forall i \ge k$ through repeated application of the equivalent control. Otherwise $\boldsymbol{x}_k \in \mathcal{X}^{\mathrm{eq}} \backslash \mathcal{S}$, but $\mathcal{X}^{\mathrm{eq}} \backslash \mathcal{S}$ is not invariant and hence $s_{k+1+i} \neq 0$ for i sufficiently large. Consequently, it can be concluded that

$$\lim_{k\to\infty} \|\boldsymbol{S}_{k+1}^\top\|_2^2 = 0. \tag{3.31}$$

and farther from (3.5), convergence of the tracking error to zero follows,

$$\lim_{k\to\infty} e_k = 0. \tag{3.32}$$

This proves the convergence part of the theorem. Finally, asymptotic stability follows from the global minimum phase property of system (3.1), (3.2) as stated in assumption 3.1. □

Remark 3.3. Note that with regard to the persistent feasibility proof, in general $\boldsymbol{U}_1^* \neq \boldsymbol{U}_1^\circ$. The control sequence chosen by the controller can certainly deviate from the sequence obtained by shifting the previous sequence and appending the equivalent control if such sequence results in a lower cost. However, the control sequence

$$\boldsymbol{U}_k = \begin{bmatrix} u_k^* & \dots & u_{k+N-1}^* & u_{k+N}^{\mathrm{eq}} \end{bmatrix} \tag{3.33}$$

analogous to (3.28) that uses the controls computed in the previous time-step along with the equivalent control law can be used to initialize numerical routines effectively.

Remark 3.4. The first inequality sign in (3.30) becomes an equality if the sequence (3.33) is applied directly, if a solution with lower associated cost is found, the inequality becomes strict.

3.4 RHSC for MIMO Regulation Problems

This section extends the developments from the previous section to multi-input multi-output systems. The fundamental ideas remain analogous to the previous section but the notation is adjusted to account for the MIMO case.

3.4.1 Terminal Invariant Set

In the MIMO case, the equivalent control, $\boldsymbol{u}_k^{\mathrm{eq}}$, is defined such that it solves the system of p algebraic equations, $\boldsymbol{s}_{k+1} = \boldsymbol{0}^{p\times 1}$, that is

$$\boldsymbol{\Psi}(\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k^{\mathrm{eq}}), \boldsymbol{r}) = \boldsymbol{0}^{p\times 1}. \tag{3.34}$$

For a given $\boldsymbol{r}$ the equivalent control depends on $\boldsymbol{x}_k$. Next, define the equivalent dynamics as

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k^{\mathrm{eq}}) =: \boldsymbol{f}^{\mathrm{eq}}(\boldsymbol{x}_k) \tag{3.35}$$

within the set

$$\mathcal{X}^{\mathrm{eq}} := \left\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{X}, \exists \boldsymbol{u}^{\mathrm{eq}} \in \mathcal{U} \text{ such that } \boldsymbol{\Psi}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}^{\mathrm{eq}}), \boldsymbol{r}) = \boldsymbol{0}^{p\times 1}\right\}. \tag{3.36}$$

The MIMO counterpart of (3.17) becomes

$$\mathcal{S} := \left\{\boldsymbol{x} : \boldsymbol{f}^{\mathrm{eq}^k}(\boldsymbol{x}) \in \mathcal{X}^{\mathrm{eq}}, \forall k \geq 0\right\}. \tag{3.37}$$

An $n - p$ dimensional subset of $\mathcal{S}$ can be found as the invariant portion of the intersection of the p sliding hypersurfaces. It reads as follows,

$$\bar{\mathcal{S}} := \left\{\boldsymbol{x} : \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{0}^{p\times 1}, \boldsymbol{f}^{\mathrm{eq}^k}(\boldsymbol{x}) \in \mathcal{X}^{\mathrm{eq}}, \forall k \geq 0\right\}. \tag{3.38}$$

3.4.2 Controller Formulation

Consider the vector sliding variable over an N-step prediction horizon for a MIMO regulation problem characterized by a constant reference, $\boldsymbol{r}_k = \boldsymbol{r}$,

$$\boldsymbol{S}_{k+1} = \begin{bmatrix} \boldsymbol{s}_{k+1} & \dots & \boldsymbol{s}_{k+N} \end{bmatrix}. \tag{3.39}$$

In (3.39), either (3.13) or (3.12) can be used as the sliding variable definition. In the first case, the following state and input sequences are obtained,

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N} \end{bmatrix}, \tag{3.40}$$
$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+N-1} \end{bmatrix} \tag{3.41}$$

and the corresponding CFTOC problem reads

$$\begin{aligned} \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} \quad & \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \boldsymbol{s}_i = \boldsymbol{\Psi}(\boldsymbol{x}_i, \boldsymbol{r}),\ i = k+1, \dots, k+N \\ & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i),\ i = k, \dots, k+N-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U},\ i = k, \dots, k+N-1 \\ & \boldsymbol{x}_{k+N} \in \mathcal{S} \\ & \boldsymbol{x}_k = \boldsymbol{x}(k). \end{aligned} \tag{3.42}$$

In the case when (3.12) is used in (3.39), the slightly longer sequences

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+\bar{N}} \end{bmatrix}, \tag{3.43}$$
$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+\bar{N}-1} \end{bmatrix} \tag{3.44}$$

are considered, where $\bar{N} = N + l$. The resulting CFTOC problem is

$$\begin{aligned} \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} \quad & \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \boldsymbol{s}_i = \mathscr{D}(\boldsymbol{e})_i, i = k+1, \dots, k+N \\ & \boldsymbol{e}_i = \boldsymbol{h}(\boldsymbol{x}_i) - \boldsymbol{r}, i = k+1, \dots, k+\bar{N} \\ & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i), i = k, \dots, k+\bar{N}-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U}, i = k, \dots, k+\bar{N}-1 \\ & \boldsymbol{x}_{k+N} \in \mathcal{S} \\ & \boldsymbol{x}_k = \boldsymbol{x}(k). \end{aligned} \tag{3.45}$$

The receding horizon sliding control strategy for the MIMO case is analogous to the SISO case and given in algorithm 3.

Algorithm 3 On-line receding horizon sliding control for nonlinear MIMO regulation problems

```
1: repeat
2:     measure $\boldsymbol{x}(k)$ at time-step $k$
3:     solve (3.42) (or (3.45)) and obtain $\boldsymbol{U}_k^*$
4:     extract first element $\boldsymbol{u}_k^*$ of $\boldsymbol{U}_k^*$ and set $\boldsymbol{u}(k) = \boldsymbol{u}_k^*$
5:     let the system evolve to $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k))$
6:     increase the time-step such that $k \leftarrow k+1$
```

3.4.3 Analysis

Below, the MIMO counterpart of the persistent feasibility and stability theorems 3.1 and 3.2 are given.

Theorem 3.3. *Given that the initial state, $\boldsymbol{x}_0 = \boldsymbol{x}(0)$, is feasible, the RHSC regulator given in algorithm 3 is persistently feasible.*

Proof. The proof is analogous to the proof of theorem 3.1 and follows from the invariance of the set $\mathcal{S}$. □

Theorem 3.4. *The output of the closed-loop system converges to its desired value, i. e. $\lim_{k\to\infty} \boldsymbol{y}_k = \boldsymbol{r}$, and the closed-loop system is asymptotically stable.*

Proof. As before, a Lyapunov-like [6] argument is utilized. Define

$$V(\boldsymbol{x}(k), \boldsymbol{U}_k) = \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2, \tag{3.46}$$

where $V^*(\boldsymbol{x}(k))$ is the optimal value and the optimizers are $\boldsymbol{X}_k^* = \begin{bmatrix} \boldsymbol{x}_k^* & \dots & \boldsymbol{x}_{k+N}^* \end{bmatrix}$ and $\boldsymbol{U}_k^* = \begin{bmatrix} \boldsymbol{u}_k^* & \dots & \boldsymbol{u}_{k+N-1}^* \end{bmatrix}$, respectively.

At the next time-step, $k+1$, the suboptimal sequence $\boldsymbol{U}_k^\circ = \begin{bmatrix} \boldsymbol{u}_{k+1}^* & \dots & \boldsymbol{u}_{k+N-1}^* & \boldsymbol{u}_{k+N}^{\mathrm{eq}} \end{bmatrix}$ yields

$$V^*(\boldsymbol{x}(k+1)) \le V(\boldsymbol{x}(k+1), \boldsymbol{U}_{k+1}^\circ) = V^*(\boldsymbol{x}(k)) - \|\boldsymbol{s}_{k+1}\|_2^2. \tag{3.47}$$

note that analogously to the SISO case $\boldsymbol{s}_{k+N+1} = \boldsymbol{0}^{p\times 1}$ and hence this term does not appear in the above equation. Since V^* is decreasing or otherwise $\boldsymbol{s}_{k+1} = \boldsymbol{0}^{p\times 1}$, it follows that

$$\lim_{k\to\infty} \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 = 0 \tag{3.48}$$

and

$$\lim_{k\to\infty} \boldsymbol{e}_k = \boldsymbol{0}^{p\times 1}. \tag{3.49}$$

In addition to convergence from (3.49), the minimum phase property from assumption 3.2 ensures that the equilibrium point associated with the steady state output, $\boldsymbol{r}$, is asymptotically stable. □

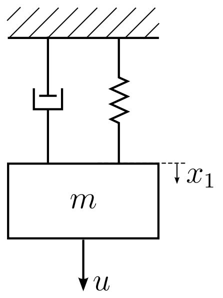

Figure 3.4: Schematic of a mass-spring-damper oscillator with spring force $c\exp(-x_1)x_1$, damping force dx_2, and control force u

3.5 Illustrative Example

To illustrate the concepts presented in this chapter, a nonlinear mass-spring-damper oscillator application is used in the following. First, the system model is introduced. Then, the RHSC design work-flow is illustrated, and finally numerical simulation results are shown.

The considered mass-spring-damper oscillator is depicted in figure 3.4 and it features a nonlinear spring stiffness. This is a popular example application used by several researchers and it was originally treated in [69], where the discrete-time state transition model was formulated as

$$\begin{bmatrix} x_{1,k+1} \\ x_{2,k+1} \end{bmatrix} = \begin{bmatrix} x_{1,k} + T_s x_{2,k} \\ x_{2,k} + \frac{T_s}{m}\left(u_k - c\exp(-x_{1,k})x_{1,k} - dx_{2,k}\right) \end{bmatrix}. \tag{3.50}$$

The nonlinear spring stiffness function is $c\exp(-x_1)$, where x_1 denotes the vertical position of the mass measured from the static equilibrium configuration. The state x_2 is the corresponding velocity. The applied control force is denoted by u and the output equation is simply

$$y_k = x_{1,k}. \tag{3.51}$$

The parameter values are $m = 0.2$, $c = 3.2$ and $d = 0.65$. The sampling time is set to $T_s = 0.1$ and the prediction horizon length is $N = 4$. The system constrains are

$$\begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \tag{3.52}$$

$$0 \leq u \leq 5. \tag{3.53}$$

Next, the RHSC design is discussed. The tracking error is simply $e_k = y_k - r$ as in (3.3), where the reference is held constant at the exemplary value $r = 0.8$. Since the relative degree of the system is $d = 2$, a sliding surface is designed using a stable first order difference operator,

$$s_k = \mathscr{D}(e)_k = \rho e_k - e_{k+1}, \tag{3.54}$$

Note that $s_k \equiv 0$ has two interpretations. Besides the interpretation as a stable difference equation, substituting for e_{k+1} using the system dynamics, yields

$$s_k = \Psi(\boldsymbol{x}_k) = \begin{bmatrix}(1-\rho) & T_s\end{bmatrix}\begin{bmatrix}x_{1,k}\\ x_{2,k}\end{bmatrix} - (1-\rho)r. \tag{3.55}$$

From (3.55) it can be seen that $s_k \equiv 0$ represents a hyperplane in x_1, x_2 space, which is the so-called sliding surface.

Next, the cost function is defined as

$$J_k = \|\boldsymbol{S}_{k+1}^\top\|_2^2 = \boldsymbol{E}_{k+1}\boldsymbol{Q}\boldsymbol{E}_{k+1}^\top. \tag{3.56}$$

Along with the standard formulation of the cost, an alternative formulation is included in (3.56). It involves

$$\boldsymbol{E}_{k+1} = \begin{bmatrix}e_{k+1} & \dots & e_{k+N+1}\end{bmatrix} \tag{3.57}$$

and $\boldsymbol{Q} \in \mathbb{R}^{(N+1)\times(N+1)}$ of the form

$$\boldsymbol{Q} = \begin{bmatrix}\rho^2 & -\rho & 0 & \dots & 0\\ -\rho & 1+\rho^2 & \ddots & \ddots & \vdots\\ 0 & \ddots & \ddots & \ddots & 0\\ \vdots & \ddots & \ddots & 1+\rho^2 & -\rho\\ 0 & \dots & 0 & -\rho & 1\end{bmatrix}. \tag{3.58}$$

This latter formulation is more in the spirit of classical model predictive control and it can be seen that the use of a sliding surface for design resulted in the introduction of off-diagonal weights. Remembering the role of ρ in (3.54), it becomes obvious that the off-diagonal weights are directly linked to stability as well as the speed of the system response. Only a single tuning parameter $0 < \rho < 1$ is involved in the controller design and it is chosen as $\rho = 0.8$.

The equivalent control law is obtained by solving (3.14). The solution is unique in this particular case and can be obtained in explicit form as

$$u_k^{\text{eq}} = \frac{m}{T_s}\left(\frac{\rho-1}{T_s}(x_{1,k} - R) + (\rho-2)x_{2,k}\right) + c\exp(-x_{1,k})x_{1,k} + dx_{2,k}. \tag{3.59}$$

The resulting equivalent dynamics from (3.15) are affine and read

$$\begin{bmatrix}x_{1,k+1}\\ x_{2,k+1}\end{bmatrix} = \begin{bmatrix}1 & T_s\\ \frac{\rho-1}{T_s} & \rho-1\end{bmatrix}\begin{bmatrix}x_{1,k}\\ x_{2,k}\end{bmatrix} + \begin{bmatrix}0\\ \frac{(1-\rho)R}{T_s}\end{bmatrix} \tag{3.60}$$

within the set $\mathcal{X}^{\text{eq}}$ as defined in (3.16).

The set $\mathcal{S}$ is the maximal positive invariant set of the autonomous system (3.60) within the set $\mathcal{X}^{\text{eq}}$. The sets $\mathcal{X}$, $\mathcal{X}^{\text{eq}}$, and $\mathcal{S}$ are shown in figure 3.5 along with the invariant part of

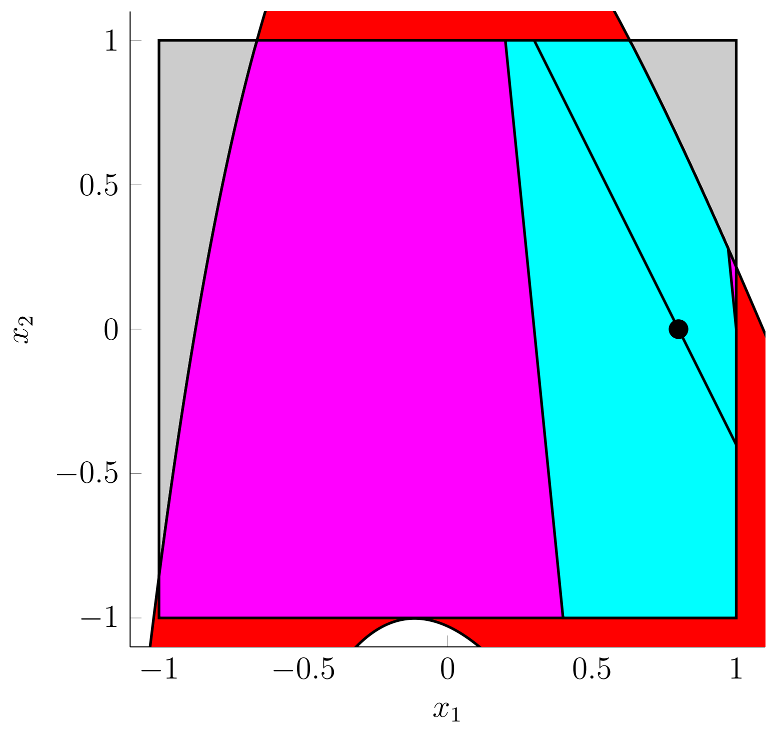

Figure 3.5: The feasible set $\mathcal{X}$ (light gray), all states where the equivalent control is feasible (red), the set $\mathcal{X}^{\text{eq}}$ (magenta), the invariant set $\mathcal{S}$ (cyan), the invariant part of the sliding surface (black, solid), and the corresponding equilibrium point (black, dot), where $\mathcal{X} \supset \mathcal{X}^{\text{eq}} \supset \mathcal{S} \supset \bar{\mathcal{S}}$

the sliding surface $\bar{\mathcal{S}}$ and the equilibrium point of (3.60). Invariant set computations were performed using the multi-parametric toolbox 3 (MPT3) in MATLAB.

Simulations were carried out in the MATLAB/SIMULINK simulation environment. The optimization problem was modeled and solved using YALMIP [68] in conjunction with the nonlinear programming solver IPOPT [97]. The output tracking performance is shown in figure 3.6. It can be seen that the output smoothly approaches the desired reference. Moreover, the figure shows that the control signal satisfies the constraints at all times indicating persistent feasibility as proven.

3.6 Concluding Remarks

This chapter reviewed different combinations of model predictive control and sliding mode control from the automatic control literature. The main contribution is the proposed receding horizon sliding control scheme, which is more widely applicable than previously proposed hybrid predictive/sliding control methods. From a mathematical point of view, RHSC is part of the class of model predictive controllers, but the underlying design process is innovative.

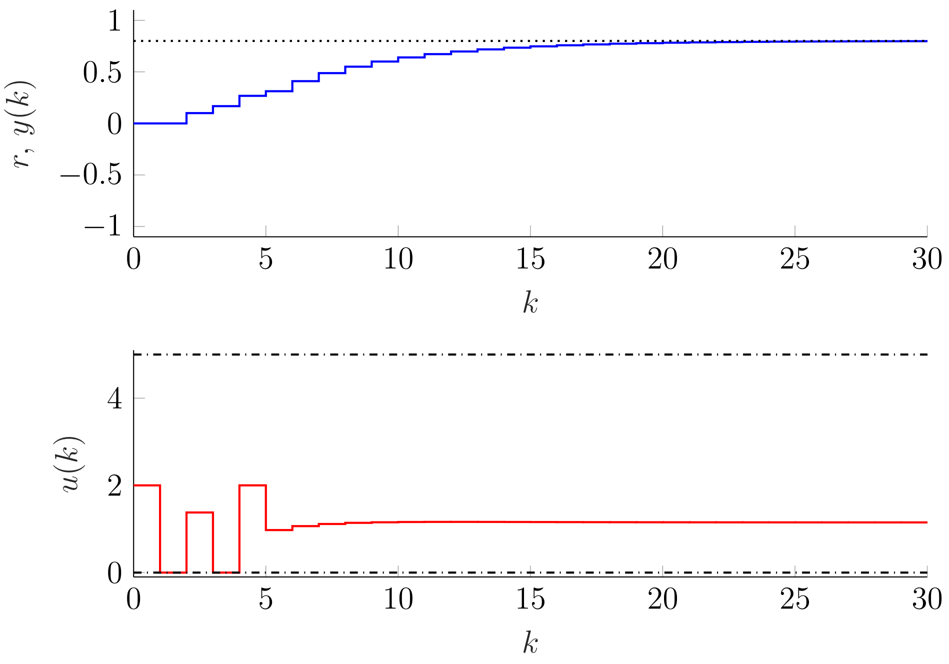

Figure 3.6: RHSC output tracking performance in terms of $y(k)$ (blue, solid) and r (black, dotted) as well as the commanded input signal $u(k)$ (red, solid) and input bounds (black, dash-dotted)

The role of a sliding surface for design is emphasized, which yields a suitable terminal invariant set and an intuitive cost function. These are the key concepts for proving the proposed method stable for a wide class of constrained nonlinear MIMO systems. From a practical point of view, the tuning is reduced to defining desired error dynamics. A simple motion control example is used to illustrate the presented ideas.

Chapter 4

Tracking RHSC for Linear Systems

This chapter adapts the receding horizon sliding control technique introduced in chapter 3 for constrained linear setpoint tracking. Section 2.3 of chapter 2 reviewed a standard approach for guaranteeing persistent feasibility of predictive controllers during setpoint changes. This approach adds an artificial reference variable, whereby allowing for reference offset at a cost specified by an additional term in the cost function. Typically, this classical approach employs a linear quadratic regulator parametrized by the artificial reference as a terminal control law and hence requires invariant set computations in an augmented state/reference space. In contrast, by exploiting the flatness property of sliding hyperplanes, this chapter eliminates the artificial reference from the control scheme. Also, the terminal invariant set is contained in the original dimensions of the state space only and hence it is generally of lower complexity. The proposed dual mode receding horizon control design approach is proven to maintain persistent feasibility and stability. This chapter has been published in slightly modified form in [40].

4.1 Motivation and Literature Review

As demonstrated in chapter 2, a key benefit of model predictive control over other control techniques is constraint satisfaction. However, this advantage comes at the expense of a challenging persistent feasibility and stability analysis. As seen in section 2.2, for regulation problems there is mature theory addressing the feasibility/stability issue by adding a suitable terminal invariant set constraint and a terminal cost term to the MPC scheme [9]. However, when target setpoint changes occur, MPC regulation schemes can still become infeasible.

The issue of changing setpoints was addressed in the literature by several authors [18, 83]. Predictive reference management for changing references is presented in [3]. A different approach is taken in [88], where an optimization variable for terminal constraint scaling is added in order to avoid infeasibility. Moreover, in a series of publications [66, 67, 23, 24] the authors develop the MPC scheme presented in section 2.3 that allows for changing setpoints by adding an additional optimization variable which has the interpretation of an artificial reference. Moreover, an additional term in the cost function that penalizes the deviation between the artificial reference and the actual reference is added. An extended terminal invariant set constraint that is formulated in terms of the augmented state including the artificial reference guarantees persistent feasibility. The authors also prove asymptotic stability with respect to feasible desired setpoints and the local optimality property [24].

Extending the state vector with the reference state is necessary in the works [66, 67, 23, 24] because terminal sets corresponding to different setpoints would overlap otherwise, which introduces ambiguity. This chapter exploits the flatness property of sliding hyperplanes to eliminate the artificial reference, which results in an invariant set solely contained in the original state space dimensions. Having a lower dimensional state space can reduce the computational burden of invariant set computations in some applications and generally reduces the number of constraints necessary to describe the set. The novel invariant set for tracking introduced in this chapter is then incorporated in a receding horizon sliding control scheme by adapting the approach from chapter 3. The controller presented in this chapter maintains all of the key properties of the MPC from [66, 67, 23, 24] and allows for eliminating the artificial reference from the control scheme resulting in a slightly reduced number of optimization variables.

The remainder of this chapter is organized as follows. In section 4.2 the concept of invariant sliding domains, i. e. invariant sets induced by sliding control laws, is derived for the tracking case. Subsequently, in section 4.3 the invariant sliding domains from section 4.2 are included in a receding horizon control framework and feasibility and stability proofs of the resulting schemes are provided. Section 4.4 contains illustrative examples and section 4.5 concludes this chapter.

4.2 Invariant Sliding Domains

It is well known that classical sliding control induces invariant sets in state space [91, 96]. This section formalizes these invariant sets in the presence of constraints. First, the considered control scenarios are presented. Then, terminal sliding control laws are derived. Finally, two invariant sets of different complexity for receding horizon control applications are defined.

4.2.1 Preliminaries

Consider square MIMO systems in state space form, where the state is denoted $\boldsymbol{x}(k) \in \mathbb{R}^n$ and $\boldsymbol{u}(k), \boldsymbol{y}(k) \in \mathbb{R}^m$ are the input and output at time-step k. Furthermore, assume an exact prediction model of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k, \tag{4.1}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k. \tag{4.2}$$

Standard notation is used, where the time-step is indicated by the subscript in order to express that equations (4.1), (4.2) yield model-based predictions of the actual system signals. The system matrices are $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times m}$, and $\boldsymbol{C} \in \mathbb{R}^{m\times n}$. Furthermore, the state and input are constrained by

$$\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n, \tag{4.3}$$

$$\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m. \tag{4.4}$$

The sets $\mathcal{X}$ and $\mathcal{U}$ are restricted to have the origin in their interior and to be polyhedral. Additionally, the following assumption is made on the system.

Assumption 4.1. The system (4.1), (4.2)

- has relative degree $(d_1, \dots, d_m)$;
- is minimum-phase, i. e. the system only has transmission zeros strictly inside the unit circle of the complex plane.

This work uses the notion of relative degree from [47] that is explicitly formalized for linear systems in [55, Definition 1]. Note that the minimum-phase assumption implies stabilizability and detectability of (4.1), (4.2) [17]. The control goal is to let the system track a piecewise constant desired output signal, $\boldsymbol{r}_k$, i. e. it is desired that the tracking error,

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \boldsymbol{r}_k, \tag{4.5}$$

is zero while accounting for the system constraints.

4.2.2 Terminal Sliding Controller

Sliding control is utilized for obtaining a terminal state feedback control law. Specifically, a Schur polynomial [56] encoding the desired eigenvalues of the output error dynamics is defined for each of the m output error components. The degree of these error dynamics is $l_i := d_i - 1$, $i = 1, \dots, m$. For invariant set computations, replace $\boldsymbol{r}_k$ in (4.5) by an artificial reference variable, $\tilde{\boldsymbol{r}}$, and form $\tilde{\boldsymbol{e}}_k = \boldsymbol{y}_k - \tilde{\boldsymbol{r}}$ [66]. With abuse of notation, the one-step ahead operator z is introduced and a sliding variable is defined by following the standard approach from [47, 89] as

$$\boldsymbol{s}_k := \begin{bmatrix} \left(\sum_{j=0}^{l_1} \alpha_{1,j} z^j\right) \tilde{e}_{1,k} \\ \vdots \\ \left(\sum_{j=0}^{l_m} \alpha_{m,j} z^j\right) \tilde{e}_{m,k} \end{bmatrix} = \begin{bmatrix} \alpha_{1,0}\tilde{e}_{1,k} + \cdots + \alpha_{1,l_1}\tilde{e}_{1,k+l_1} \\ \vdots \\ \alpha_{m,0}\tilde{e}_{m,k} + \cdots + \alpha_{m,l_m}\tilde{e}_{m,k+l_m} \end{bmatrix}. \tag{4.6}$$

There are two design restrictions. Firstly, for stability reasons the polynomials $\sum_{j=0}^{l_i} \alpha_{i,j} z^j$ with $i = 1, \dots, m$ are required to be chosen such that all roots are strictly inside of the unit circle. Secondly, it is required that $\alpha_{i,l_i} \neq 0, i = 1, \dots, m$. Farther, normalize without loss of generality such that $\alpha_{i,l_i} = 1, i = 1, \dots, m$.

By recursively substituting the system model (4.1), (4.2) in (4.6) the following representation for $\boldsymbol{s}_k$ results,

$$\boldsymbol{s}_k = \begin{bmatrix} \boldsymbol{c}_1^\top \sum_{j=0}^{l_1} \alpha_{1,j} \boldsymbol{A}^j \\ \vdots \\ \boldsymbol{c}_m^\top \sum_{j=0}^{l_m} \alpha_{m,j} \boldsymbol{A}^j \end{bmatrix} \boldsymbol{x}_k - \begin{bmatrix} \sum_{j=0}^{l_1} \alpha_{1,j} & & \\ & \ddots & \\ & & \sum_{j=0}^{l_m} \alpha_{m,j} \end{bmatrix} \tilde{\boldsymbol{r}} =: \boldsymbol{G}\boldsymbol{x}_k - \boldsymbol{H}\tilde{\boldsymbol{r}}. \tag{4.7}$$

The form of $\boldsymbol{s}_k$ is standard, compare [92], with an added constant involving the reference to account for reference tracking rather than regulation to the origin. In (4.7), all off-diagonal entries of $\boldsymbol{H}$ are zero and $\boldsymbol{c}_1^\top, \dots, \boldsymbol{c}_m^\top$ denote the rows of $\boldsymbol{C}$. It is important to note that the specified design restrictions enforce $\boldsymbol{c}_i^\top \boldsymbol{A}^j \boldsymbol{B} = \boldsymbol{0}^{1\times m}$, $j = 0, \dots, l_i$, $i = 1, \dots, m$ and hence these terms do not feature in (4.7).

Given a certain reference $\tilde{\boldsymbol{r}}$, from (4.7) it is easy to interpret $\{\boldsymbol{x} : \boldsymbol{s} = \boldsymbol{0}^{m\times 1}\}$ as the intersection of m hyperplanes in state space. On this intersection, the specified desired error dynamics $\alpha_{i,0}\tilde{e}_{i,k} + \cdots + \alpha_{i,l_i}\tilde{e}_{i,k+l_i} = 0, i = 1, \dots, m$ hold. For $\{\boldsymbol{x} : \boldsymbol{s} \neq \boldsymbol{0}^{m\times 1}\}$, the components s_i are a measure for the distance of $\boldsymbol{x}$ to the desired manifolds $\{\boldsymbol{x}^* : \boldsymbol{g}_i^\top \boldsymbol{x}^* - H_{i,i}\tilde{r}_i = 0\}$ which is directly quantified by $s_i / \|\boldsymbol{g}_i\|_2, i = 1, \dots, m$, where $\boldsymbol{g}_i^\top$ is the i^{th} row of $\boldsymbol{G}$.

The so-called equivalent control law [96, 92], that enforces the state to be on the sliding manifold in the subsequent time-step is found by setting $\boldsymbol{s}_{k+1} = \boldsymbol{0}^{m\times 1}$. Using (4.7) it results that

$$\boldsymbol{u}_k^{\text{eq}} = -\left(\boldsymbol{G}\boldsymbol{B}\right)^{-1}\left(\boldsymbol{G}\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{H}\tilde{\boldsymbol{r}}\right) =: \boldsymbol{K}\boldsymbol{x}_k + \boldsymbol{L}\tilde{\boldsymbol{r}}, \tag{4.8}$$

where existence of $(\boldsymbol{G}\boldsymbol{B})^{-1}$ is ensured by assumption 4.1 [55].

4.2.3 Invariant Sliding Domain for Tracking

Next, adapt the procedure from [66] for computing an invariant set for tracking when (4.8) is used as a terminal controller. Therefore, extend the state vector to $\boldsymbol{w} = \begin{bmatrix}\boldsymbol{x}^\top & \tilde{\boldsymbol{r}}^\top\end{bmatrix}^\top$. The closed-loop dynamics obtained by application of (4.8) are

$$\boldsymbol{w}_{k+1} = \begin{bmatrix} \boldsymbol{A} + \boldsymbol{B}\boldsymbol{K} & \boldsymbol{B}\boldsymbol{L} \\ \boldsymbol{0}^{m\times n} & \boldsymbol{I}^{m\times m} \end{bmatrix} \boldsymbol{w}_k =: \boldsymbol{A}^{\text{eq}} \boldsymbol{w}_k \tag{4.9}$$

subject to the polyhedral constraint $\boldsymbol{w} \in \mathcal{W}^{\text{eq}}$, where

$$\mathcal{W}^{\text{eq}} := \left\{\boldsymbol{w} : \begin{bmatrix}\boldsymbol{I}^{n\times n} & \boldsymbol{0}^{n\times m}\end{bmatrix} \boldsymbol{w} \in \mathcal{X},\ \begin{bmatrix}\boldsymbol{K} & \boldsymbol{L}\end{bmatrix} \boldsymbol{w} \in \mathcal{U}\right\}. \tag{4.10}$$

Note that the matrix $\boldsymbol{A}+\boldsymbol{BK}$ has all eigenvalues strictly inside the unit disc. In particular, the gain matrix $\boldsymbol{K}$ places m eigenvalues at zero, $\sum_{i=1}^{m} l_i$ eigenvalues come from the design of (4.6), and the remaining eigenvalues, if any, are given by the transmission zeros and by the uncontrollable or unobservable eigenvalues of $\boldsymbol{A}$, all of which are strictly stable as a result of assumption 4.1. Following [66], the invariant set for tracking is

$$\mathcal{T} := \left\{\boldsymbol{w} : \boldsymbol{A}^{\mathrm{eq}^k}\boldsymbol{w} \in \mathcal{W}^{\mathrm{eq}}, \forall k \geq 0\right\}. \tag{4.11}$$

In the remainder, the notation $\mathcal{S} := \mathrm{proj}_{\boldsymbol{x}}(\mathcal{T})$ is occasionally used for the projection of the invariant sliding domain $\mathcal{T}$ on the original state space.

4.2.4 Reduced Invariant Sliding Domain for Tracking

A subset of $\mathcal{S}$ can be obtained with a lower complexity procedure compared to the approach from section 4.2.3 by exploiting the following result.

Proposition 4.1. *For every state $\boldsymbol{x}_k$ there exists exactly one artificial reference value $\tilde{\boldsymbol{r}}$, such that $\boldsymbol{x}_k$ lies in the affine space $\left\{\boldsymbol{x} : \boldsymbol{s} = \boldsymbol{0}^{m\times 1}\right\}$ with $\boldsymbol{s}$ parametrized by $\tilde{\boldsymbol{r}}$.*

Proof. By construction, exactly one $\tilde{\boldsymbol{r}}$ is found for every $\boldsymbol{x}_k$

$$\boldsymbol{s}_k = \boldsymbol{0}^{m\times 1} \quad \Leftrightarrow \quad \tilde{\boldsymbol{r}} = \boldsymbol{H}^{-1}\boldsymbol{G}\boldsymbol{x}_k. \tag{4.12}$$

It was used that the matrix $\boldsymbol{H}^{-1}$ always exists. This follows from the diagonal structure of $\boldsymbol{H}$ as seen in (4.7) and the fact that the polynomials $\sum_{j=0}^{l_i}\alpha_{i,j}z^j, i = 1,\ldots,m$ were chosen to have no roots on or outside the unit circle. Hence $z = 1$ cannot be a root and $\sum_{j=0}^{l_i}\alpha_{i,j} \neq 0, i = 1,\ldots,m$. □

Substituting from (4.12) in (4.8) yields an alternative form of the equivalent control law. Instead of driving the system to a surface parametrized by a specific $\tilde{\boldsymbol{r}}$, this version of the equivalent control keeps the state at time $k+1$ on exactly the same sliding manifold that it is associated with at time k. In other words, set $\boldsymbol{s}_{k+1} = \boldsymbol{s}_k$ and find

$$\bar{\boldsymbol{u}}_k^{\mathrm{eq}} = -\left(\boldsymbol{GB}\right)^{-1}\boldsymbol{G}\left(\boldsymbol{A} - \boldsymbol{I}^{n\times n}\right)\boldsymbol{x}_k =: \bar{\boldsymbol{K}}\boldsymbol{x}_k. \tag{4.13}$$

Then, substituting (4.13) in (4.1) yields the equivalent dynamics

$$\boldsymbol{x}_{k+1} = \left(\boldsymbol{A} + \boldsymbol{B}\bar{\boldsymbol{K}}\right)\boldsymbol{x}_k =: \bar{\boldsymbol{A}}^{\mathrm{eq}}\boldsymbol{x}_k, \tag{4.14}$$

which are constrained by $\boldsymbol{x} \in \bar{\mathcal{X}}^{\mathrm{eq}}$ with

$$\bar{\mathcal{X}}^{\mathrm{eq}} := \left\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{X}, \bar{\boldsymbol{K}}\boldsymbol{x} \in \mathcal{U}\right\}. \tag{4.15}$$

Finally, the reduced invariant sliding domain is defined as

$$\bar{\mathcal{S}} := \left\{\boldsymbol{x} : \bar{\boldsymbol{A}}^{\mathrm{eq}^k}\boldsymbol{x} \in \bar{\mathcal{X}}^{\mathrm{eq}}, \forall k \geq 0\right\}. \tag{4.16}$$

Remark 4.1. Computing $\mathcal{T}$ and $\bar{\mathcal{S}}$ requires invariant set computations for autonomous systems subject to polyhedral constraints, which can be done with open source tools such as the MPT3 for MATLAB [41]. An advantage of the reduced invariant sliding domain over the original version is that its computation can be performed in $\mathbb{R}^n$ rather than $\mathbb{R}^{n+m}$ and hence lower computational complexity is generally expected. A disadvantage is that the proposed method only renders a subset of $\mathcal{S}$ invariant, i. e. $\bar{\mathcal{S}} \subseteq \mathcal{S}$.

4.2.5 Set of Equilibrium Points

Finally, also define the set of all feasible steady states [66]. Therefore, let $\boldsymbol{q} = \begin{bmatrix} \boldsymbol{x}^\top & \boldsymbol{u}^\top & \boldsymbol{y}^\top \end{bmatrix}^\top$ and

$$\begin{aligned}\mathcal{Q}^{\text{ss}} := \Bigg\{ \boldsymbol{q} : \begin{bmatrix} \boldsymbol{A} - \boldsymbol{I}^{n\times n} & \boldsymbol{B} & \boldsymbol{0}^{n\times m} \\ \boldsymbol{C} & \boldsymbol{0}^{m\times m} & -\boldsymbol{I}^{m\times m} \end{bmatrix} \boldsymbol{q} = \begin{bmatrix} \boldsymbol{0}^{n\times 1} \\ \boldsymbol{0}^{m\times 1} \end{bmatrix}, \begin{bmatrix} \boldsymbol{I}^{n\times n} & \boldsymbol{0}^{n\times 2m} \end{bmatrix} \boldsymbol{q} \in \mathcal{X}, \\ \begin{bmatrix} \boldsymbol{0}^{m\times n} & \boldsymbol{I}^{m\times m} & \boldsymbol{0}^{m\times m} \end{bmatrix} \boldsymbol{q} \in \mathcal{U} \Bigg\}. \end{aligned} \tag{4.17}$$

The sets of feasible steady states, inputs, and outputs can be obtained through projection as $\mathcal{X}^{\text{ss}} := \text{proj}_{\boldsymbol{x}}(\mathcal{Q}^{\text{ss}})$, $\mathcal{U}^{\text{ss}} := \text{proj}_{\boldsymbol{u}}(\mathcal{Q}^{\text{ss}})$, and $\mathcal{Y}^{\text{ss}} := \text{proj}_{\boldsymbol{y}}(\mathcal{Q}^{\text{ss}})$.

Given an arbitrary target output $\tilde{\boldsymbol{r}} \in \mathcal{Y}^{\text{ss}}$, the associated steady state and input are uniquely determined as

$$\begin{bmatrix} \tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{u}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} - \boldsymbol{I}^{n\times n} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{0}^{m\times m} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{0}^{n\times 1} \\ \tilde{\boldsymbol{r}} \end{bmatrix} =: \begin{bmatrix} \boldsymbol{N} \\ \boldsymbol{M} \end{bmatrix} \tilde{\boldsymbol{r}}. \tag{4.18}$$

The matrix inverse in (4.18) always exists for square systems satisfying assumption 4.1 [17].

4.3 Receding Horizon Sliding Tracking Control

This section incorporates invariant sliding domains in receding horizon tracking control schemes originally proposed in [39]. First the controller formulations are presented and subsequently feasibility and stability theorems follow.

4.3.1 Controller Formulations

Consider the quantity from (4.7) over an N-step prediction horizon,

$$\boldsymbol{S}_{k+1} := \begin{bmatrix} \boldsymbol{s}_{k+1} & \dots & \boldsymbol{s}_{k+N} \end{bmatrix}, \tag{4.19}$$

and rewrite state and input sequences similarly as

$$\boldsymbol{X}_k := \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N} \end{bmatrix}, \tag{4.20}$$

$$\boldsymbol{U}_k := \begin{bmatrix} \boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+N-1} \end{bmatrix}. \tag{4.21}$$

Next, formulate the following optimization problem representing a constrained finite-time optimal control problem

$$\begin{aligned}\min_{\boldsymbol{X}_k,\boldsymbol{U}_k,\tilde{\boldsymbol{r}}_k} \quad & \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 + \|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_k - \boldsymbol{r}_k\right)\|_1 \\ \text{s. t.} \quad & \boldsymbol{s}_i = \boldsymbol{G}\boldsymbol{x}_i - \boldsymbol{H}\tilde{\boldsymbol{r}}_k,\ i = k+1,\ldots,k+N \\ & \boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i,\ i = k,\ldots,k+N-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U},\ i = k,\ldots,k+N-1 \\ & \begin{bmatrix}\boldsymbol{x}_{k+N}^\top & \tilde{\boldsymbol{r}}_k^\top\end{bmatrix}^\top \in \mathcal{T} \\ & \boldsymbol{x}_k = \boldsymbol{x}(k).\end{aligned} \tag{4.22}$$

In (4.22), $\boldsymbol{S}_{k+1}$ is minimized in the Frobenius norm and an offset cost $\|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_k - \boldsymbol{r}_k\right)\|_1$ penalizes deviations of $\tilde{\boldsymbol{r}}_k$ from the desired $\boldsymbol{r}_k$.

An alternative optimization problem can be obtained by eliminating the artificial reference from (4.22) with the relation $\tilde{\boldsymbol{r}}_k = \boldsymbol{H}^{-1}\boldsymbol{G}\boldsymbol{x}_{k+N}$. Note that (4.23) does not contain $\tilde{\boldsymbol{r}}_k$,

$$\begin{aligned}\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \quad & \|\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 + \|\boldsymbol{T}\left(\boldsymbol{H}^{-1}\boldsymbol{G}\boldsymbol{x}_{k+N} - \boldsymbol{r}_k\right)\|_1 \\ \text{s. t.} \quad & \boldsymbol{s}_i = \boldsymbol{G}\left(\boldsymbol{x}_i - \boldsymbol{x}_{k+N}\right),\ i = k+1,\ldots,k+N \\ & \boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i,\ i = k,\ldots,k+N-1 \\ & \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{u}_i \in \mathcal{U},\ i = k,\ldots,k+N-1 \\ & \boldsymbol{x}_{k+N} \in \bar{\mathcal{S}} \\ & \boldsymbol{x}_k = \boldsymbol{x}(k).\end{aligned} \tag{4.23}$$

The L^1 norm is used for the offset cost in (4.22) and (4.23) to recover the local optimality property [24].

Finally, a controller is obtained based on the standard receding horizon control principle as summarized in algorithm 4 [9]. At every time-step, a sequence of future control actions is obtained by solving the optimization problem (4.22) or (4.23), respectively. The first element of the resulting control sequence is commanded to the plant and the procedure repeats at the next time-step.

Algorithm 4 Online receding horizon sliding tracking control for linear systems

```
1: repeat
2:     measure $\boldsymbol{x}(k)$ at time-step $k$
3:     solve (4.23) (or (4.22)) and obtain $\boldsymbol{U}_k^*$
4:     extract first column $\boldsymbol{u}_k^*$ of $\boldsymbol{U}_k^*$ and set $\boldsymbol{u}(k) = \boldsymbol{u}_k^*$
5:     let the system evolve to $\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k)$
6:     increase the time-step such that $k \leftarrow k+1$
```

Remark 4.2. Both optimization problems, (4.22) and (4.23), can be formulated as quadratic programs [10]. If $\boldsymbol{T} = \boldsymbol{H}$ is chosen in (4.23), then the offset cost becomes $\|\boldsymbol{G}\boldsymbol{x}_{k+N} - \boldsymbol{H}\boldsymbol{r}_k\|_1$. This offset cost penalizes the distance of the terminal state from the desired intersection of m sliding hyperplanes parametrized by the actual reference. Another sensible choice for $\boldsymbol{T}$ is the identity matrix.

4.3.2 Analysis

The standard assumption 4.2 ensures feasibility at the initial time-step. Then, feasibility for all future time-steps follows in theorem 4.1.

Assumption 4.2. The initial state $\boldsymbol{x}(0)$ is such that (4.22) (or (4.23)) is feasible.

Theorem 4.1. *Algorithm 4 is persistently feasible.*

Proof. The invariance property of the terminal sets $\mathcal{T}$ and $\bar{\mathcal{S}}$ immediately yields persistent feasibility for the receding horizon control schemes. See [9] for further details. $\square$

Remark 4.3. Since $\bar{\mathcal{S}} \subseteq \mathcal{S}$, the feasibility domain for (4.22) is generally larger than that of (4.23). Note that increasing N by 1 in scheme (4.23) results in both schemes having the same feasibility set because for the target set $\bar{\mathcal{S}}$ the one-step controllable set of system (4.1), (4.2) subject to state and input constraints [9] is actually given by $\mathcal{S}$, i. e. $\mathcal{S} = \mathcal{K}_1(\bar{\mathcal{S}})$.

Next, turn to the stability analysis. The stability proof relies on the following intermediate result.

Lemma 4.1 (From [66])**.** *Let $\hat{\boldsymbol{r}}$ be a feasible target reference and $\hat{\boldsymbol{x}} = \boldsymbol{N}\hat{\boldsymbol{r}}$, $\hat{\boldsymbol{u}} = \boldsymbol{M}\hat{\boldsymbol{r}}$ are the corresponding target steady state and input. Let $\tilde{\boldsymbol{x}} = \boldsymbol{N}\tilde{\boldsymbol{r}} \in \mathcal{X}^{\mathrm{ss}}$ and $\tilde{\boldsymbol{u}} = \boldsymbol{M}\tilde{\boldsymbol{r}} \in \mathcal{U}^{\mathrm{ss}}$ be a steady-state and input with $\tilde{\boldsymbol{x}} \in \mathrm{int}(\mathcal{X})$, $\tilde{\boldsymbol{u}} \in \mathrm{int}(\mathcal{U})$. Then there exists a $\lambda \in [0,1)$ and a corresponding reference $\boldsymbol{r}' = \lambda\tilde{\boldsymbol{r}} + (1-\lambda)\hat{\boldsymbol{r}}$ such that $\tilde{\boldsymbol{x}}$ satisfies $\tilde{\boldsymbol{x}} \in \mathcal{T}_{|\boldsymbol{r}'} := \left\{\boldsymbol{x} : \begin{bmatrix}\boldsymbol{x}^\top & \boldsymbol{r}'^\top\end{bmatrix}^\top \in \mathcal{T}\right\}$.*

Proof. Pick an arbitrary positive definite $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and denote the unique positive definite solution of the Lyapunov equation $(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{K})^\top \boldsymbol{P}(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{K}) - \boldsymbol{P} = -\boldsymbol{Q}$ as $\boldsymbol{P} \in \mathbb{R}^{n\times n}$. Let a class of convex sublevel sets be denoted by $\mathcal{E}(\boldsymbol{x}_0, \epsilon) := \left\{\boldsymbol{x} : (\boldsymbol{x}-\boldsymbol{x}_0)^\top \boldsymbol{P}(\boldsymbol{x}-\boldsymbol{x}_0) \le \epsilon\right\}$. Furthermore, denote by $\boldsymbol{x}' = \boldsymbol{N}\boldsymbol{r}'$ and $\boldsymbol{u}' = \boldsymbol{M}\boldsymbol{r}'$ the steady state and input corresponding to $\boldsymbol{r}'$.

Given that $\tilde{\boldsymbol{x}} \in \mathrm{int}(\mathcal{X})$, $\tilde{\boldsymbol{u}} \in \mathrm{int}(\mathcal{U})$ lie in the interior of the feasible set, there exist constants $\beta > 0$ and $\gamma \in (0,1)$ such that $\boldsymbol{x} \in \gamma\mathcal{X}$ and $\boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\tilde{\boldsymbol{r}} \in \gamma\mathcal{U}$ for all $\boldsymbol{x}$ in a neighborhood of $\tilde{\boldsymbol{x}}$ namely $\boldsymbol{x} \in \mathcal{E}(\tilde{\boldsymbol{x}}, \beta)$. Choose $\lambda \in [0,1)$ sufficiently close to 1 such that $\boldsymbol{L}(\boldsymbol{r}' - \tilde{\boldsymbol{r}}) \in (1-\gamma)\mathcal{U}$ and such that there exists a $\delta > 0$ with $\tilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{x}', \delta) \subset \mathcal{E}(\tilde{\boldsymbol{x}}, \beta)$. Finally, find that for all $\boldsymbol{x} \in \mathcal{E}(\boldsymbol{x}', \delta)$ it results that $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\boldsymbol{r}' = \boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\tilde{\boldsymbol{r}} + \boldsymbol{L}(\boldsymbol{r}' - \tilde{\boldsymbol{r}}) \in \mathcal{U}$. This proves $\tilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{x}', \delta) \subset \mathcal{T}_{|\boldsymbol{r}'}$.

$\square$

Theorem 4.2. *Let $\boldsymbol{r}_k$ be constant for all $k \geq \bar{k}$. Then algorithm 4 asymptotically drives the closed-loop system to the feasible steady state output that minimizes the offset cost.*

Proof. The proof follows the idea to show that the cost function is a Lyapunov-like function [6]. Assume $k \geq \bar{k}$ and let $\boldsymbol{r}_k = \boldsymbol{r}_\infty$ be arbitrary but constant. Denote the cost function from (4.22) by

$$V(\boldsymbol{x}(k), \boldsymbol{r}_\infty, \boldsymbol{U}_k, \tilde{\boldsymbol{r}}_k) := \|\boldsymbol{S}_{k+1}\|_\mathrm{F}^2 + \|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_k - \boldsymbol{r}_\infty\right)\|_1 \tag{4.24}$$

and its optimal value by $V^*(\boldsymbol{x}(k), \boldsymbol{r}_\infty)$. The corresponding minimizing arguments are denoted by the symbols $\boldsymbol{X}_k^* := \begin{bmatrix}\boldsymbol{x}_k^* & \dots & \boldsymbol{x}_{k+N}^*\end{bmatrix}$, $\boldsymbol{U}_k^* := \begin{bmatrix}\boldsymbol{u}_k^* & \dots & \boldsymbol{u}_{k+N-1}^*\end{bmatrix}$, and $\tilde{\boldsymbol{r}}_k^*$.

At the subsequent time-step $k+1$, consider the feasible but not necessarily optimal control sequence $\boldsymbol{U}_{k+1}^\circ = \begin{bmatrix}\boldsymbol{u}_{k+1}^* & \dots & \boldsymbol{u}_{k+N-1}^* & \boldsymbol{K}\boldsymbol{x}_{k+N}^* + \boldsymbol{L}\tilde{\boldsymbol{r}}_k^*\end{bmatrix}$ and the artificial reference $\tilde{\boldsymbol{r}}_{k+1}^\circ = \tilde{\boldsymbol{r}}_k^*$. Then, it results that

$$V^*(\boldsymbol{x}(k+1), \boldsymbol{r}_\infty) \leq V(\boldsymbol{x}(k+1), \boldsymbol{r}_\infty, \boldsymbol{U}_{k+1}^\circ, \tilde{\boldsymbol{r}}_{k+1}^\circ) = V^*(\boldsymbol{x}(k), \boldsymbol{r}_\infty) - \|\boldsymbol{s}_{k+1}\|_2^2. \tag{4.25}$$

Inequality (4.25) implies that V^* is strictly decreasing from time-step k to $k+1$ unless $\boldsymbol{s}_{k+1} = \boldsymbol{0}^{m\times 1}$. But $\boldsymbol{s}_{k+1} = \boldsymbol{0}^{m\times 1}$ implies that the equivalent control $\boldsymbol{u}(k) = \boldsymbol{K}\boldsymbol{x}(k) + \boldsymbol{L}\tilde{\boldsymbol{r}}_k^*$ is applied and hence exponential convergence $\boldsymbol{x}(k) \to \tilde{\boldsymbol{x}} = \boldsymbol{N}\tilde{\boldsymbol{r}}_k^*$ follows. Hence, after elapsing a finite number of time-steps, Δ, the system enters a neighborhood of $\tilde{\boldsymbol{x}}$, where lemma 1 applies and $\boldsymbol{x}(\kappa) \in \mathcal{T}_{|\boldsymbol{r}'}$ with $\kappa := k + \Delta$. Then, using the feasible but not necessarily optimal sequence $\bar{\boldsymbol{U}}_{\kappa+1}^\circ$ consisting of repeatedly applying the equivalent control law $\boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\boldsymbol{r}'$ and choosing $\tilde{\boldsymbol{r}}_{\kappa+1}^\circ = \boldsymbol{r}'$ it follows that

$$\begin{aligned}V^*(\boldsymbol{x}(\kappa+1), \boldsymbol{r}_\infty) &\leq V(\boldsymbol{x}(\kappa+1), \boldsymbol{r}_\infty, \bar{\boldsymbol{U}}_{\kappa+1}^\circ, \tilde{\boldsymbol{r}}_{\kappa+1}^\circ)\\ &= V^*(\boldsymbol{x}(\kappa), \boldsymbol{r}_\infty) + \|\boldsymbol{T}\left(\boldsymbol{r}' - \boldsymbol{r}_\infty\right)\|_1 - \|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_k^* - \boldsymbol{r}_\infty\right)\|_1\\ &< V^*(\boldsymbol{x}(\kappa), \boldsymbol{r}_\infty).\end{aligned} \tag{4.26}$$

Hence, conclude that

$$\lim_{k\to\infty} \|\boldsymbol{S}_{k+1}\|_\mathrm{F}^2 = 0 \tag{4.27}$$

and the artificial setpoint converges to the closest value within the set of feasible outputs,

$$\lim_{k\to\infty} \|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_k - \boldsymbol{r}_\infty\right)\|_1 = \min_{\tilde{\boldsymbol{r}}_\infty \in \mathcal{Y}^\mathrm{ss}} \|\boldsymbol{T}\left(\tilde{\boldsymbol{r}}_\infty - \boldsymbol{r}_\infty\right)\|_1. \tag{4.28}$$

Consequently, referring back to (4.6), the result (4.27), (4.28) yields $\lim_{k\to\infty} \boldsymbol{y}_k - \tilde{\boldsymbol{r}}_\infty = \boldsymbol{0}^{m\times 1}$ and asymptotic convergence of the tracking error to any feasible reference setpoint can be concluded.

Notice that (4.23) can be obtained from (4.22) by adding the constraint $\boldsymbol{s}_{k+N} = \boldsymbol{0}^{m\times 1} \Leftrightarrow \tilde{\boldsymbol{r}}_k = \boldsymbol{H}^{-1}\boldsymbol{G}\boldsymbol{x}_{k+N}$ which also yields $\left\{\boldsymbol{x} : \begin{bmatrix}\boldsymbol{x}^\top & \tilde{\boldsymbol{r}}^\top\end{bmatrix}^\top \in \mathcal{T}, \tilde{\boldsymbol{r}} = \boldsymbol{H}^{-1}\boldsymbol{G}\boldsymbol{x}\right\} = \bar{\mathcal{S}}$. The stability proof remains unchanged by adding the constraint $\boldsymbol{s}_{k+N} = \boldsymbol{0}^{m\times 1}$, hence the stability argument for scheme (4.22) also applies for (4.23). □

4.4 Illustrative Examples

This section contains two examples. One example applies the proposed method to a single-input single-output system and the other example considers a multi-input multi-output system.

4.4.1 A SISO System

As a first example to illustrate the ideas presented in this chapter, consider the SISO system

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} 0.7 & 0.5 \\ -0.8 & 1.6 \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{x}_k + \underbrace{\begin{bmatrix} 0 \\ 0.7 \end{bmatrix}}_{\boldsymbol{B}} u_k, \tag{4.29}$$

$$y_k = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{\boldsymbol{C}} \boldsymbol{x}_k. \tag{4.30}$$

Since $\boldsymbol{u}$ and $\boldsymbol{y}$ are scalar in this scenario, they are denoted by u and y, respectively. The sets of feasible states and inputs are

$$\mathcal{X} = \left\{ \boldsymbol{x} : \begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq \boldsymbol{x} \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}, \tag{4.31}$$

$$\mathcal{U} = \{ u : -1 \leq u \leq 1 \}. \tag{4.32}$$

Since the relative degree of y is $d = 2$, the sliding variable is designed as

$$s_k = -\rho \tilde{e}_k + \tilde{e}_{k+1}, \tag{4.33}$$

where the component indices have been dropped because all quantities are scalar. The tuning parameter ρ is selected to be $\rho = 0.6$. The quantities $\boldsymbol{G}$ and $\boldsymbol{H}$ in (4.7) can be evaluated as

$$\boldsymbol{G} = \begin{bmatrix} 0.1 & 0.5 \end{bmatrix}, \quad H = 0.4, \tag{4.34}$$

where H is used to denote the quantity $\boldsymbol{H}$ because it is scalar in this particular example.

The computed sets $\mathcal{T}$ and $\bar{\mathcal{S}}$ are shown in figures 4.1 and 4.2, respectively. The sets were computed with the MPT3 [41] in MATLAB. Besides the set $\mathcal{T}$, figure 4.1 also shows slices of $\mathcal{T}$, the corresponding sliding surfaces, and equilibrium points for three exemplary values of the desired reference. On the other hand, figure 4.2 shows the sets $\bar{\mathcal{S}}$ and $\mathcal{S}$ along with sliding surfaces and equilibrium points for the same exemplary reference values. Comparing figures 4.1 and 4.2, the dimensionality reduction obtained with the developed method is obvious as the set $\mathcal{T}$ is a subset of $\mathbb{R}^3$ and described by 12 inequalities or 20 vertices whereas $\bar{\mathcal{S}}$ is a subset of $\mathcal{R}^2$ and described by 6 inequalities or 6 vertices.

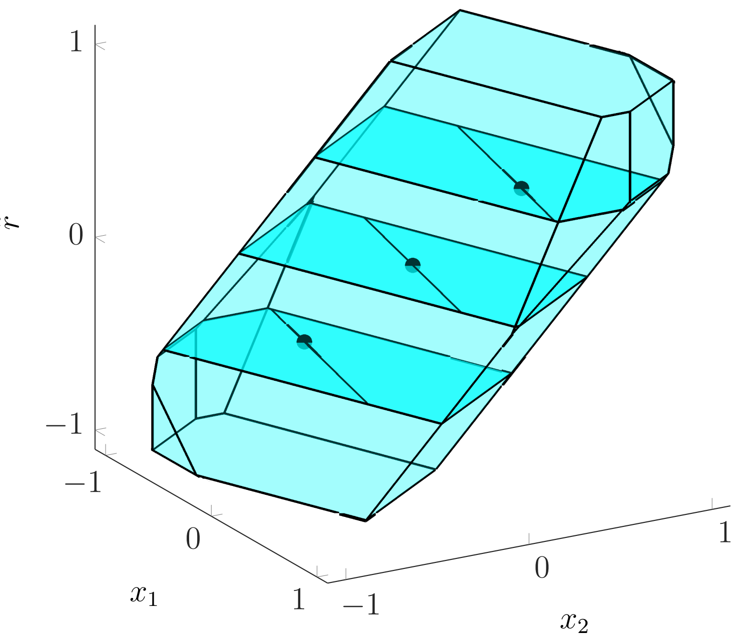

Figure 4.1: The set $\mathcal{T} \subset \mathbb{R}^3$ (light cyan) as well as slices of $\mathcal{T}$ (dark cyan), sliding surfaces (black, solid), and equilibrium points (black, dot) for the desired output values 0 and ± 0.5

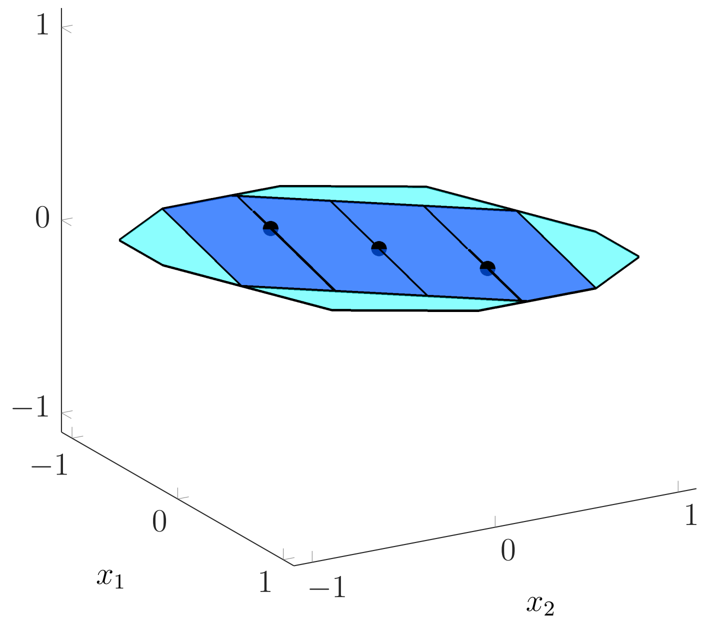

Figure 4.2: The sets $\bar{\mathcal{S}} \subset \mathbb{R}^2$ (blue) and $\mathcal{S} \subset \mathbb{R}^2$ (cyan) with $\bar{\mathcal{S}} \subset \mathcal{S}$ as well as sliding surfaces (black, solid), and equilibrium points (black, dot) for the desired output values 0 and ± 0.5

Closed-loop simulations of control algorithm 4 with both (4.23) and (4.22) in line 3 were completed. MATLAB/SIMULINK was used as the simulation environment and YALMIP [68] and GUROBI [36] were utilized for formulating and repeatedly solving the optimization problems involved in the respective controller formulation. The horizon length was set to $N = 5$ steps. The reference is designed to drive the system output to the exemplary values that were also used in figures 4.1 and 4.2, namely 0 and ± 0.5. Tracking results are shown in figure 4.3 and the associated control signals are plotted in figure 4.4. The simulation confirms that the closed-loop control system remains persistently feasible and asymptotically stable with respect to the given setpoints. Furthermore, the results from both control algorithms are virtually identical, indicating that the complexity reduction obtained with the proposed approach of using (4.23) in line 3 of algorithm 4 does not lead to a noticeable performance degradation in this particular example.

4.4.2 A MIMO System

As a second example, consider the unstable system

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0.9 & -1.6 & 0.3 \\ 0.4 & -0.2 & 1.5 \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{x}_k + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\boldsymbol{B}} \boldsymbol{u}_k, \tag{4.35}$$

$$\boldsymbol{y}_k = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{C}} \boldsymbol{x}_k. \tag{4.36}$$

The sets of feasible states and inputs are

$$\mathcal{X} = \left\{ \boldsymbol{x} : \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \leq \boldsymbol{x} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}, \tag{4.37}$$

$$\mathcal{U} = \left\{ \boldsymbol{u} : \begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq \boldsymbol{u} \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}. \tag{4.38}$$

Since the relative degrees of y_1 and y_2 are $d_1 = 2$ and $d_2 = 1$, respectively, design the sliding variables as

$$s_{1,k} = -\rho \tilde{e}_{1,k} + \tilde{e}_{1,k+1}, \tag{4.39}$$

$$s_{2,k} = \tilde{e}_{2,k} \tag{4.40}$$

with $\rho = 0.3$. Then the matrices $\boldsymbol{G}$ and $\boldsymbol{H}$ in (4.7) become

$$\boldsymbol{G} = \begin{bmatrix} -0.3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{H} = \begin{bmatrix} 0.7 & 0 \\ 0 & 1 \end{bmatrix}. \tag{4.41}$$

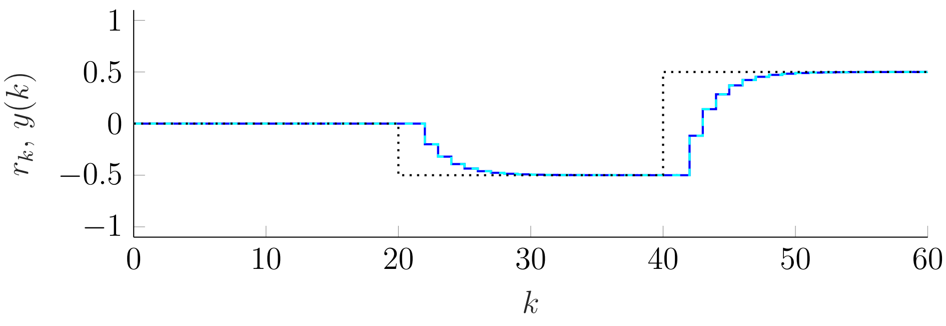

Figure 4.3: Tracking performance in terms of the desired reference r_k (black, dotted) and the controlled output $y(k)$ when using algorithm 4 with (4.23) (blue, solid) and (4.22) (cyan, dashed) in line 3

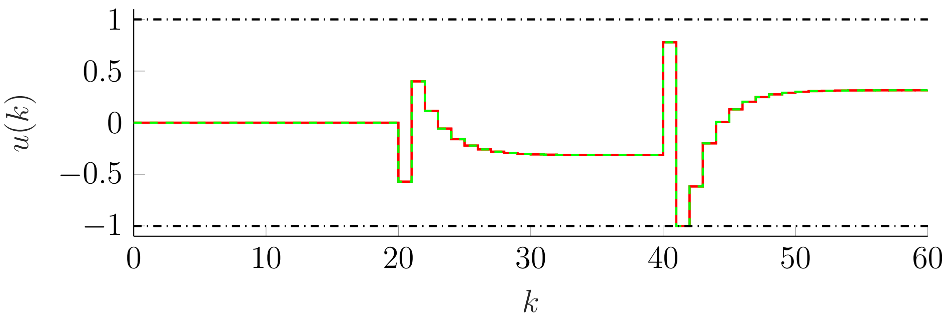

Figure 4.4: Control signal $u(k)$ obtained with algorithm 4 with (4.23) (red, solid) and (4.22) (green, dashed) in line 3 as well as bounds on the control signals (black, dash-dotted)

The sets $\mathcal{S}$ and $\bar{\mathcal{S}}$ as defined in sections 4.2.3 and 4.2.4 are again computed with the MPT3 in MATLAB and are plotted in figure 4.5 along with $\mathcal{X}$ and $\mathcal{X}^{\text{ss}}$. It can be clearly seen that $\mathcal{S} \supset \bar{\mathcal{S}}$, indicating that the control scheme (4.22) will have a larger feasibility set compared to scheme (4.23). The set $\bar{\mathcal{S}}$ is determined by 8 constraints, on the other hand, the set $\mathcal{T}$ requires a larger number of 16 constraints.

Closed-loop simulations are performed using control algorithm 4 with (4.23) in line 3. The simulations were performed in MATLAB/SIMULINK using YALMIP [68] and GUROBI [36] for formulating and repeatedly solving optimization problem (4.23). The horizon length was set to $N = 10$ steps. For an exemplary reference signal, the controller's tracking performance is shown in figure 4.6 and the corresponding control signals are shown in figure 4.7. The simulation demonstrates that the closed-loop system remains persistently feasible and that it is asymptotically stable with respect to feasible setpoints. For infeasible setpoints, the state converges to the best feasible setpoint as seen between $k = 50$ and $k = 75$, where an input constraint is active. In conclusion, simulations confirm the result that the artificial reference can safely be removed by utilizing the method proposed in this chapter.

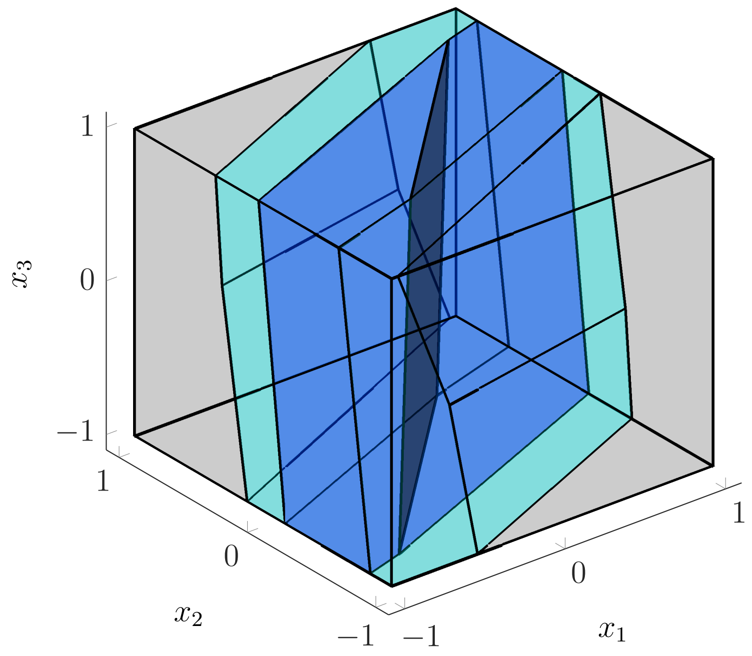

Figure 4.5: Feasible set, $\mathcal{X}$ (light gray), invariant sliding domains, $\mathcal{S}$ (cyan) and $\bar{\mathcal{S}}$ (blue), as well as the set of feasible steady states $\mathcal{X}^{\text{ss}}$ (dark gray) with $\mathcal{X} \supset \mathcal{S} \supset \bar{\mathcal{S}} \supset \mathcal{X}^{\text{ss}}$

4.5 Concluding Remarks

This chapter has developed receding horizon sliding control for setpoint tracking. In particular, for square linear MIMO systems it is shown that the invariant sliding domain is a suitable terminal invariant set for receding horizon tracking control. By exploiting the flatness property of sliding surfaces, an approach of lower complexity when compared to classical methods in the literature can be derived. It is proven that the proposed dual-mode control scheme maintains the persistent feasibility and stability properties. Examples illustrate the simplicity and effectiveness of the proposed approach.

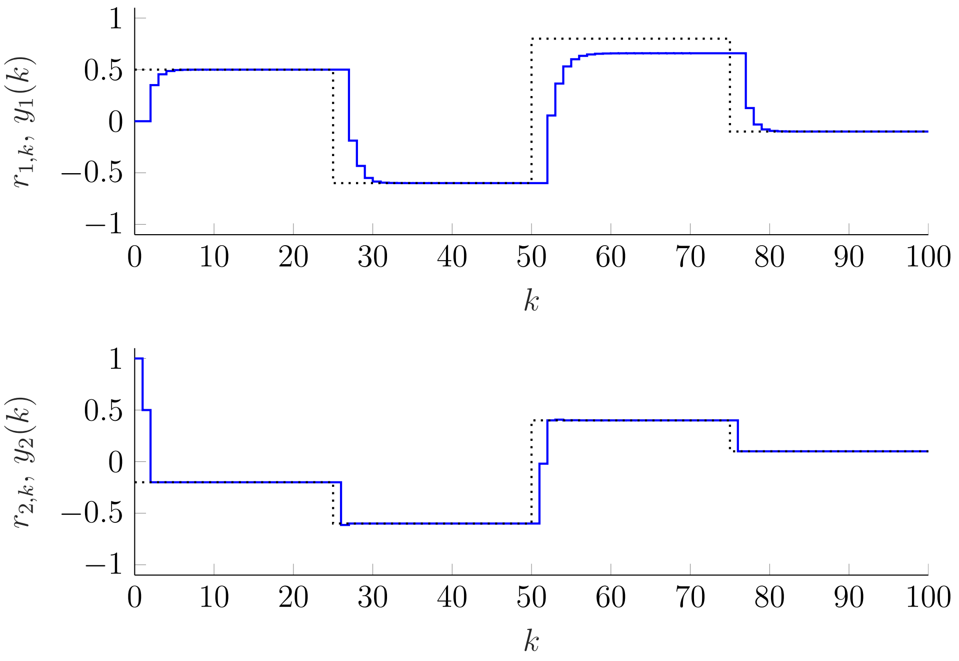

Figure 4.6: Tracking performance of algorithm 4 with (4.23) in line 3 in terms of the desired reference components $r_{1,k}$, $r_{2,k}$ (black, dotted) and the controlled outputs $y_1(k)$, $y_2(k)$ (blue, solid)

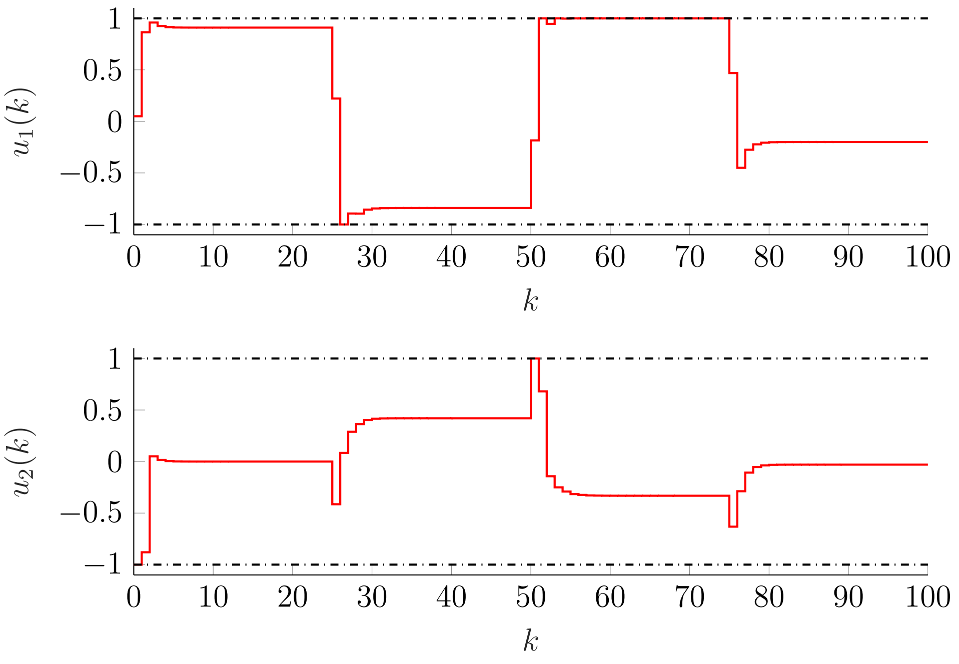

Figure 4.7: Control signal components $u_1(k)$, $u_2(k)$ (red, solid) obtained with algorithm 4 with (4.23) in line 3 and bounds on the control signals (black, dash-dotted)

Part III

Applications

Chapter 5

Path Tracking with Micro-Underwater Vehicles

Autonomous underwater vehicles (AUVs) are advancing the state of the art in numerous scientific and commercial aquatic applications including oceanography and environmental monitoring. The current surge in micro-aerial vehicles enables the development of low-cost, small micro AUVs and it is expected that underwater vehicles will gain increasing popularity in the near future. Given the high potential of AUVs, in this chapter a waypoint path following autopilot for underwater vehicles is developed utilizing the receding horizon sliding control method proposed in this dissertation. It is shown that the receding horizon sliding control cost function is an adequate choice from a practical point of view. In particular, the presented results confirm that the designed algorithm yields excellent tracking performance while accounting for actuator constraints on the AUV's propellers. Moreover, the system is found to be robust with respect to the most common sources of uncertainty in underwater robotics such as uncertain hydrodynamic forces and localization errors.

5.1 Motivation

Over the time span of the past ten years, the deployment of autonomous underwater vehicles has rapidly increased. Popular applications for AUVs include environmental sampling, seafloor mapping, monitoring of offshore sites, as well as exploration. The current surge in AUVs is directly linked to recent advances in the area of micro-aerial vehicles, which led to a miniaturization and cost decrease of hardware, such as motor controllers and sensor kits. Beyond single vehicle operation, the recent trends also enable the development of micro AUVs for multi-vehicle operations, e. g. the HippoCampus vehicle from [37]. Figure 5.1 depicts a computer-aided design (CAD) drawing of HippoCampus.

One of the most important tasks for AUV operations is path following. The challenge of the path following control problem for AUVs is due to the underactuated, nonlinear, and uncertain system dynamics [59, 5]. Moreover, since micro AUVs are increasingly popular for

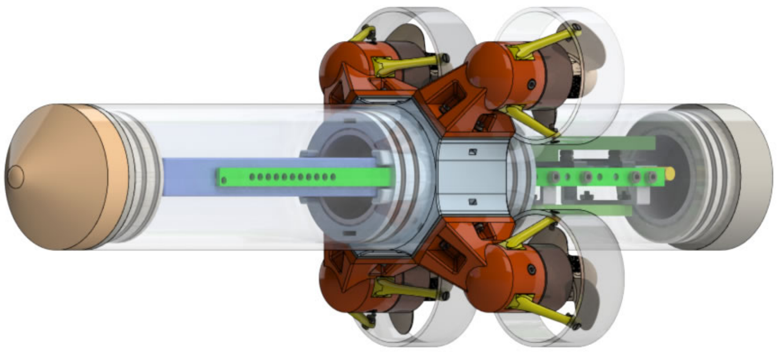

Figure 5.1: CAD drawing of the HippoCampus [37] micro-autonomous underwater vehicle

operations in confined environments such as nuclear storage ponds [35], path following systems have to enable highly agile AUV maneuvering. This motivates research on autonomous waypoint path following for underactuated agile AUVs.

A common architecture for path tracking systems assumes that a path planning algorithm generates a set of waypoints depending on the specific AUV application at hand. Then, by connecting these waypoints with line segments a path is derived and fed into a path tracking control system. Unlike trajectory tracking, path following does not include time constraints [74]. Different control methodologies can be applied to achieve path following, e. g. explicit feedback-based control or model predictive control. The line of sight (LOS) method from [26] is a widely used feedback-based control framework for path following. The LOS method is based on a geometric relationship where a LOS vector from the vehicle position to the path is defined. Using the LOS vector a desired reference can be computed and tracked using an explicit feedback controller. Despite the fact that LOS methods have been studied extensively in the last ten years and have been successfully applied to 3 degree of freedom (DOF) underactuated surface vessels [7, 62], contributions in this area rarely consider the full underactuated AUV dynamics for waypoint path following. An alternative approach that is often used instead decomposes the LOS path following problem in space into a vertical and a horizontal part, where a 3 DOF system is assumed [61] in either plane. In [8], a complete description of the AUV dynamics is considered for guiding an AUV along a single line path. While the model is similar to the one presented in this work, no abrupt directional changes of the path are considered in [8] which usually pose a great challenge in path following, especially in confined test tanks.

MPC based methods explicitly allow for the incorporation of physical constraints such as actuator limits. Furthermore, MPC's anticipatory behavior allows accounting for time delays and future errors which is beneficial for achieving agile vehicle motion. Different MPC implementations have been reported for marine vessels, mostly for surface vessels with planar dynamics [72] or for AUVs with decoupled dynamics [93]. MPC has also been applied to station keeping operations of underwater vehicles [22]. However, overall MPC based path following approaches for AUVs operating in all three dimensions are still rare.

As a case study, this chapter designs an RHSC for waypoint tracking of micro AUVs. The result is a unified approach for waypoint path following in 3-D for underactuated AUVs and the presented control framework is in principle suitable to run on resource constrained micro AUVs. As in most complex applications, the invariant set constraint is not used and stability is concluded from simulations. Furthermore, uncertainties are often ignored in the classical MPC design phase due to computational complexity issues. Instead, one relies on the capability of the nominal controller to manage disturbances, i. e. one uses a property often referred to as the inherent robustness of the controller [1]. As pointed out in [32], MPC usually achieves higher levels of inherent robustness, when changes in the control signal are penalized. However, high input rate penalties also yield a sluggish controller that does not fully exploit the actuator potentials. Note that RHSC bypasses the tradeoff between control effort and tracking accuracy by introducing desired error dynamics. This chapter shows that the error dynamics involved in the RHSC design can be tuned to yield an inherently robust controller without restricting control action at all. Hence, this chapter confirms that the RHSC cost function is not only theoretically sound but that it can also be a sensible choice from a practical point of view.

The remainder of this chapter is organized as follows. Section 5.2 introduces the AUV model utilized in this work. Receding horizon sliding control design for the particular application at hand is carried out in section 5.3. Section 5.4 presents numerical experiments of the closed-loop control system. Closing remarks are given in section 5.5.

5.2 Modeling

This section introduces an AUV model that is used for control design [25] in latter parts of this chapter. Define $\boldsymbol{\eta} = \begin{bmatrix} X & Y & Z & \epsilon_0 & \epsilon_1 & \epsilon_2 & \epsilon_3 \end{bmatrix}^\top$ to describe position and orientation with respect to an inertial frame, where a unit quaternion $\boldsymbol{q} = \begin{bmatrix} \epsilon_0 & \epsilon_1 & \epsilon_2 & \epsilon_3 \end{bmatrix}^\top$ is used to parametrize orientation in order to avoid gimbal lock. Furthermore, let $\boldsymbol{\nu} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^\top$ denote translational and angular velocities in the body fixed frame.

The AUV dynamics equations can be formulated as

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu}, \tag{5.1}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{M}_A\dot{\boldsymbol{\nu}} + \boldsymbol{C}_A(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}_A(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{k}, \tag{5.2}$$

where $\boldsymbol{J}$ is the transformation matrix between body fixed and inertial reference frames, $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{M}_A$ is the hydrodynamic added mass matrix, $\boldsymbol{D}_A$ is the hydrodynamic damping matrix, $\boldsymbol{C}$ and $\boldsymbol{C}_A$ are the rigid body and hydrodynamic Coriolis and centripetal matrix, respectively. The sum $\boldsymbol{M}_A\dot{\boldsymbol{\nu}} + \boldsymbol{C}_A(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}_A(\boldsymbol{\nu})\boldsymbol{\nu}$ collectively models hydrodynamic loads. The weight and hydrostatic loads are considered through $\boldsymbol{g}(\boldsymbol{\eta})$, which depends on the vehicle orientation. The force vector $\boldsymbol{k}$ includes the system inputs for thrust, roll, pitch, as well as yaw and it reads $\boldsymbol{k} = (\boldsymbol{M} + \boldsymbol{M}_A)\begin{bmatrix} \tau_{\dot{x}} & 0 & 0 & \tau_{\dot{\phi}} & \tau_{\dot{\theta}} & \tau_{\dot{\psi}} \end{bmatrix}^\top$. It is assumed that the motor controllers are able to drive the motors to generate the desired thrust and moments.

Note that all quantities are understood to be measured in SI units and hence units are dropped in the remainder of this chapter.

The matrix $\boldsymbol{J}(\boldsymbol{\eta})$ involved in equation (5.1) relates quantities in the inertial and body fixed reference frames and it is given by

$$\boldsymbol{J}(\boldsymbol{\eta}) = \begin{bmatrix} \boldsymbol{J}_1(\boldsymbol{\eta}) & \boldsymbol{0}^{3\times 3} \\ \boldsymbol{0}^{4\times 3} & \boldsymbol{J}_2(\boldsymbol{\eta}) \end{bmatrix}, \tag{5.3}$$

where

$$\boldsymbol{J}_1(\boldsymbol{\eta}) = \begin{bmatrix} 1-2(\epsilon_2^2+\epsilon_3^2) & 2(\epsilon_1\epsilon_2-\epsilon_3\epsilon_0) & 2(\epsilon_1\epsilon_3+\epsilon_2\epsilon_0) \\ 2(\epsilon_1\epsilon_2+\epsilon_3\epsilon_0) & 1-2(\epsilon_1^2+\epsilon_3^2) & 2(\epsilon_2\epsilon_3-\epsilon_1\epsilon_0) \\ 2(\epsilon_1\epsilon_3-\epsilon_2\epsilon_0) & 2(\epsilon_2\epsilon_3+\epsilon_1\epsilon_0) & 1-2(\epsilon_1^2+\epsilon_2^2) \end{bmatrix}, \tag{5.4}$$

$$\boldsymbol{J}_2(\boldsymbol{\eta}) = \frac{1}{2}\begin{bmatrix} -\epsilon_1 & -\epsilon_2 & -\epsilon_3 \\ \epsilon_0 & -\epsilon_3 & \epsilon_2 \\ \epsilon_3 & \epsilon_0 & -\epsilon_1 \\ -\epsilon_2 & \epsilon_1 & \epsilon_0 \end{bmatrix}. \tag{5.5}$$

By rearranging equation (5.2), the equations of motion can be derived in compact non-linear state space form as

$$\dot{\boldsymbol{x}} = \bar{\boldsymbol{f}}_c(\boldsymbol{x}, \bar{\boldsymbol{u}}), \tag{5.6}$$

where $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{\eta}^\top & \boldsymbol{\nu}^\top\end{bmatrix}^\top$ is the state vector and $\bar{\boldsymbol{u}} = \begin{bmatrix}\tau_{\dot{x}} & \tau_{\dot{\phi}} & \tau_{\dot{\theta}} & \tau_{\dot{\psi}}\end{bmatrix}^\top$ contains the input variables.

5.3 Control System Design

In this section the path following control system is developed. First, the control objective is defined. Subsequently, auxiliary control laws for roll angle and speed error regulation are given. Finally, the core path following logic is synthesized based on the RHSC method.

5.3.1 Control Objective

Given a set of waypoints for navigation purposes, define a path by connecting successive waypoints, e. g. $\boldsymbol{w}_j$ and $\boldsymbol{w}_{j+1}$, with a line segment denoted l_j as illustrated in figure 5.2. The cross-track error e_j is the distance between the closest line segment l_j and the center of mass of the AUV, $\boldsymbol{p} = \begin{bmatrix}X & Y & Z\end{bmatrix}^\top$, within the plane perpendicular to the path. Since the path is non-smooth and the vehicle dynamics underactuated, it is important to clarify the desired path following objective. Either tracking of the Dubins path, i. e. exact passing through all waypoints, can be seeked or the cross-track error can be minimized while approaching the waypoints as closely as possible, which is referred to as the approximating path in [60]. The focus of this work is on approximating paths. For a given set of waypoints the control objective is to minimize e_j while maintaining a prescribed surge velocity.

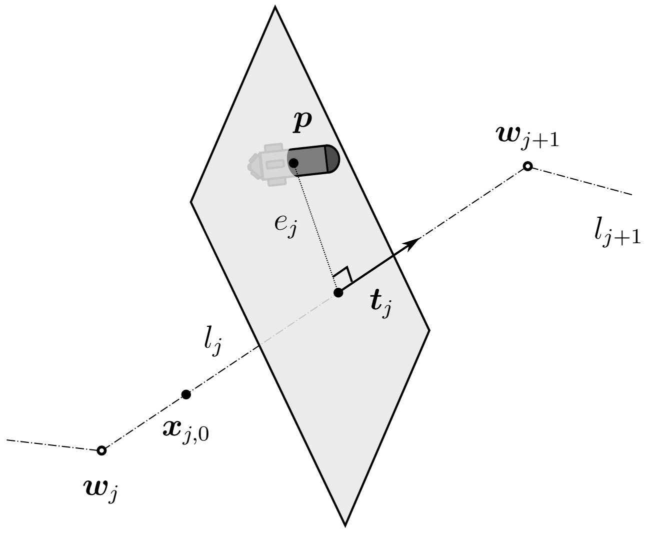

Figure 5.2: Waypoint-based path definition and parametrization of the cross-track error

As a preliminary for expressing the cross-track error with respect to the j^{th} line segment, a line segment's unit tangent vector is introduced as

$$\boldsymbol{t}_j = \frac{\boldsymbol{w}_{j+1} - \boldsymbol{w}_j}{\|\boldsymbol{w}_{j+1} - \boldsymbol{w}_j\|_2}. \tag{5.7}$$

Then, exploiting the geometric relations involved in defining the cross track error, it follows that

$$e_j = \|(\boldsymbol{p} - \boldsymbol{x}_{j,0}) \times \boldsymbol{t}_j\|_2 \,, \tag{5.8}$$

where $\boldsymbol{x}_{j,0}$ is an arbitrary point on the line segment l_j. Since the unit tangent vector, $\boldsymbol{t}_j$, is constant along each line segment the cross-track error can be evaluated relatively efficiently within an optimization routine.

5.3.2 Speed Error and Roll Regulation

In addition to a desired waypoint path, a prescribed surge velocity is assumed to be given for executing the desired maneuver. Moreover, notice that the system model is symmetric with respect to the longitudinal vehicle axis. The roll angle introduces arbitrariness in the control problem, hence, the roll angle is desired to vanish when possible. The AUV plant is augmented with proportional (P) and proportional-derivative (PD) feedback control loops for speed error and roll angle regulation

$$\tau_{\dot{x}} = k_{p,\dot{x}}(\dot{x} - \dot{x}^{\text{des}}), \tag{5.9}$$

$$\tau_{\dot{\phi}} = k_{p,\dot{\phi}}\phi + k_{d,\dot{\phi}}\dot{\phi}. \tag{5.10}$$

The parameters $k_{p,\dot{x}}$, $k_{p,\dot{\phi}}$, $k_{d,\dot{\phi}}$ are the control gains. In (5.9), $\dot{x}^{\text{des}}$ represents the desired vehicle speed. The roll angle ϕ in (5.10) can be recovered from the quaternion representation using the four quadrant inverse tangent function,

$$\phi = \operatorname{atan2}\left(2(\epsilon_2\epsilon_3 + \epsilon_1\epsilon_0)), 1 - 2(\epsilon_1^2 + \epsilon_2^2)\right). \tag{5.11}$$

A manipulated system model follows by substituting (5.9), (5.10) in (5.6) and can be written as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_c(\boldsymbol{x}, \boldsymbol{u}). \tag{5.12}$$

The model (5.12) is parametrized by $\dot{x}^{\text{des}}$ as well as the control gains. The remaining control inputs for navigating the AUV are the yaw and pitch moments, i. e. $\boldsymbol{u} = \begin{bmatrix}\tau_{\dot{\theta}} & \tau_{\dot{\psi}}\end{bmatrix}^\top$. The receding horizon sliding control strategy will be used for computing these inputs.

5.3.3 RHSC Steering Algorithm

The receding horizon sliding control framework used in this work was developed in chapter 3. The method builds on a discrete-time representation of the system dynamics, which is obtained from (5.12) through Euler discretization [77],

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k). \tag{5.13}$$

The time-step is indicated by subscripts, representing the system's signals at times kT_s and $(k+1)T_s$, respectively, where T_s is the sampling time.

Further following chapter 3, the first step of the RHSC design procedure is the definition of a sliding variable [39]. Notice that the relative degree of the cross track error from (5.8) is equal to two, which is a common characteristic of mechanical systems. Hence, a first order difference operator is applied to $e_{j,k}$ in order to obtain a sliding variable as

$$s_k = e_{j,k+1} - \rho e_{j,k} =: \mathscr{D}(e)_k. \tag{5.14}$$

The tuning parameter ρ is chosen such that $0 < \rho < 1$. Intuitively, the control goal is now reduced to driving s_k to zero and exploiting the fact that desired error dynamics are resulting when $s_k \equiv 0$. The index j in (5.14) corresponds to the line segment that is closest to the AUV position at time-step k.

Formally, RHSC minimizes s_k over an N-step prediction horizon. For notational simplicity, define

$$\boldsymbol{S}_{k+1} = \begin{bmatrix}s_{k+1} & \dots & s_{k+N}\end{bmatrix}. \tag{5.15}$$

Likewise, the state and control vector sequences are abbreviated as follows,

$$\boldsymbol{X}_k = \begin{bmatrix}\boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N+1}\end{bmatrix}, \tag{5.16}$$

$$\boldsymbol{U}_k = \begin{bmatrix}\boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+N}\end{bmatrix}. \tag{5.17}$$

The explicit form of the CFTOC problem, that the RHSC solves at every sampling-instant reads as follows,

$$\min_{\boldsymbol{X}_k, \boldsymbol{U}_k} \quad \|\boldsymbol{S}_{k+1}^{\top}\|_2^2 \tag{5.18}$$

$$\begin{aligned} \text{s. t.} \quad & s_i = \mathscr{D}(e_j)_i, \ i = k, \dots, k+N \\ & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i), \ i = k, \dots, k+N \\ & \boldsymbol{u}_\mathrm{l} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_\mathrm{u}, \ i = k, \dots, k+N \\ & \boldsymbol{x}_k = \boldsymbol{x}(k). \end{aligned} \tag{5.19}$$

The problem is subject to the system dynamics and input constraints. Lower and upper bounds on the controls are denoted by $\boldsymbol{u}_\mathrm{l}$ and $\boldsymbol{u}_\mathrm{u}$. The problem is initialized with the current state that is assumed to be measurable. Once problem (5.18) is solved, the first element of the solution sequence is applied to the AUV and the same problem is solved after letting time elapse and reaching the next sampling-instant.

5.4 Results

This section presents the utilized simulation setup and evaluates the control system performance based on the obtained numerical results.

5.4.1 Simulation Setup

The closed-loop control system is simulated in MATLAB/SIMULINK. The controller is written in a C code-based S-function involving the nonlinear programming solver NPSOL [34]. The sampling time of the RHSC is set to $T_s = 100\,\mathrm{ms}$ and the prediction horizon length is $N = 8$ steps. The plant dynamics are integrated with a 10 ms step-length Runge-Kutta 4 scheme emulating continuous time. The low complexity P and PD control laws are also updated every 10 ms. Path following in $\mathbb{R}^3$ is considered, where the path is defined by straight line segments that connect waypoints. The RHSC is analyzed in two different scenarios. First, a nominal case is considered, where the controller uses the exact model parameters and the discretization inaccuracy involved in finding (5.13) from (5.12) is the only considered uncertainty source. Second, a perturbed scenario is considered, where the controller model differs significantly from the simulation model. In the perturbed case, the controller model is such that the hydrodynamic added Coriolis matrix $\boldsymbol{C}_A$ vanishes and the hydrodynamic damping $\boldsymbol{D}_A$ has an error of 50%. Furthermore, the position measurement is corrupted by zero-mean white Gaussian measurement noise with a standard deviation of 0.1 m which is a realistic assumption in confined test tanks [33]. Also, actuator bounds were multiplied by a factor of 2.5 for the nominal case and are kept realistic in the perturbed case.

Identifier	A	B	C	D	E
Location	$\begin{bmatrix}1 & 1 & 0\end{bmatrix}^\top$	$\begin{bmatrix}9 & 9 & 0\end{bmatrix}^\top$	$\begin{bmatrix}10 & 10 & 4\end{bmatrix}^\top$	$\begin{bmatrix}17 & 10 & 5\end{bmatrix}^\top$	$\begin{bmatrix}18 & 9 & 8\end{bmatrix}^\top$

Table 5.1: Waypoint identifiers and locations for numerical simulation scenarios

5.4.2 Results

A sample path is defined by five waypoints A, B, C, D, E with coordinates as listed in table 5.1. The chosen system parameters correspond to the ones of the HippoCampus micro AUV shown in figure 5.1 [37]. The dry mass of the robot is 2 kg and the mass matrix reads

$$\boldsymbol{M} = \mathrm{diag}(2, 2, 2, 0.1, 0.4, 0.4).$$

The hydrodynamic added mass is

$$\boldsymbol{M}_A = \mathrm{diag}(1, 2, 2, 0.1, 0.5, 0.5)$$

and the hydrodynamic added damping is

$$\boldsymbol{D}_A = \mathrm{diag}(2, 5, 5, 3, 8, 8).$$

Cross-diagonal damping elements are not included, because additional entries in $\boldsymbol{D}_A$ will lead to more stable dynamics due to the passivity of damping. The hydrodynamic Coriolis matrix is chosen to be

$$\boldsymbol{C}_A = \begin{bmatrix} 0 & 0 & 0 & 0 & -2\dot{z} & 2\dot{y} \\ 0 & 0 & 0 & 2\dot{z} & 0 & -0.4\dot{x} \\ 0 & 0 & 0 & -2\dot{y} & 0.4\dot{x} & 0 \\ 0 & -2\dot{z} & 2\dot{y} & 0 & -0.5\dot{\psi} & 0.5\dot{\theta} \\ 2\dot{z} & 0 & -0.4\dot{x} & 0.5\dot{\psi} & 0 & -0.1\dot{\phi} \\ -2\dot{y} & 0.4\dot{x} & 0 & -0.5\dot{\theta} & 0.1\dot{\phi} & 0 \end{bmatrix}. \tag{5.20}$$

The micro AUV starts at rest at the initial position $\begin{bmatrix}1 & 0 & 0\end{bmatrix}^\top$. Simulations are carried out with a total simulation time length of 30 seconds.

Five different values for the tuning parameter ρ are considered and the root mean square tracking error for all five cases are given in table 5.2 for the nominal and perturbed scenario. For the nominal case, the table shows that a smaller ρ value leads to smaller root mean square errors. This is expected since a smaller ρ implies faster desired error convergence and a more aggressive controller. Furthermore, comparing the root mean square error of the nominal and perturbed simulations for $\rho = 0.975$, $\rho = 0.95$, $\rho = 0.9$ unveils that the controller performance is only slightly affected by the introduced uncertainty and noise. However, for smaller ρ values, the AUV fails at completing the path following task, i. e. the AUV does not reach segment D–E within the 30 second simulation time. This indicates that

Tuning	$\rho = 0.975$	$\rho = 0.95$	$\rho = 0.9$	$\rho = 0.8$	$\rho = 0.6$
Nominal case	0.3020	0.2404	0.1976	0.1877	0.1793
Perturbed case	0.3096	0.2522	0.2034	fail	fail

Table 5.2: Root mean square cross track errors from simulations of the nominal and the perturbed closed-loop system; failure means that the AUV did not reach segment D–E within the simulation time window due to swerving motions

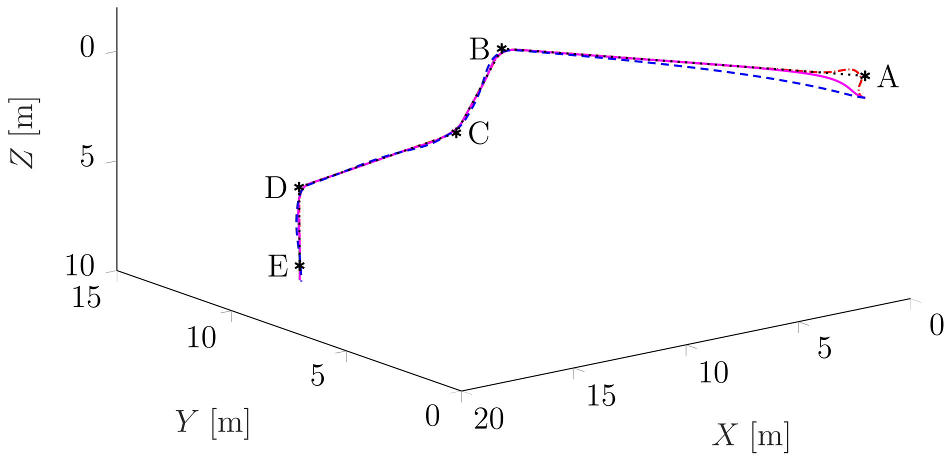

Figure 5.3: Simulated position of AUV's center of mass and the desired path (black, dotted) defined by waypoints in 3-D for nominal case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted); the initial vehicle position at rest is $X = 1$, $Y = 0$, $Z = 0$

the controller is rather robust with respect to noise as long as gains are not tuned to be exceedingly aggressive.

More details on the simulations are given next for the tuning values $\rho = 0.975$, $\rho = 0.9$, $\rho = 0.6$. For the nominal case, figure 5.3 shows the path tracking performance in space for the three selected parameter values. It can be seen that the controller succeeds at stabilizing the AUV on the desired path for all tuning values. As expected, the error decreases faster when ρ is chosen smaller which is also confirmed by figure 5.4, where the simulated cross-track errors are shown. Note, since the path is defined by waypoints connected with line segments and the cross-track error is computed with regard to line segments, jumps in the cross-track error are unavoidable. This is a direct consequence of the so-called approximating path following approach described in section 5.3.1, for which the vehicle trajectory does not necessarily pass through the waypoints. The computed control actions in terms of the yaw

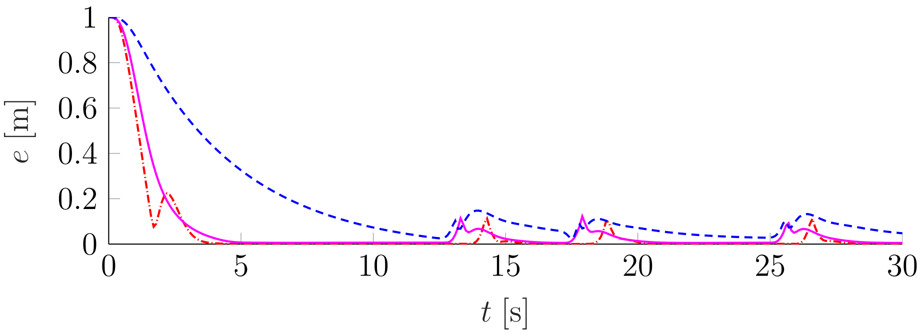

Figure 5.4: Cross-track error of the AUV over simulation time for the nominal case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

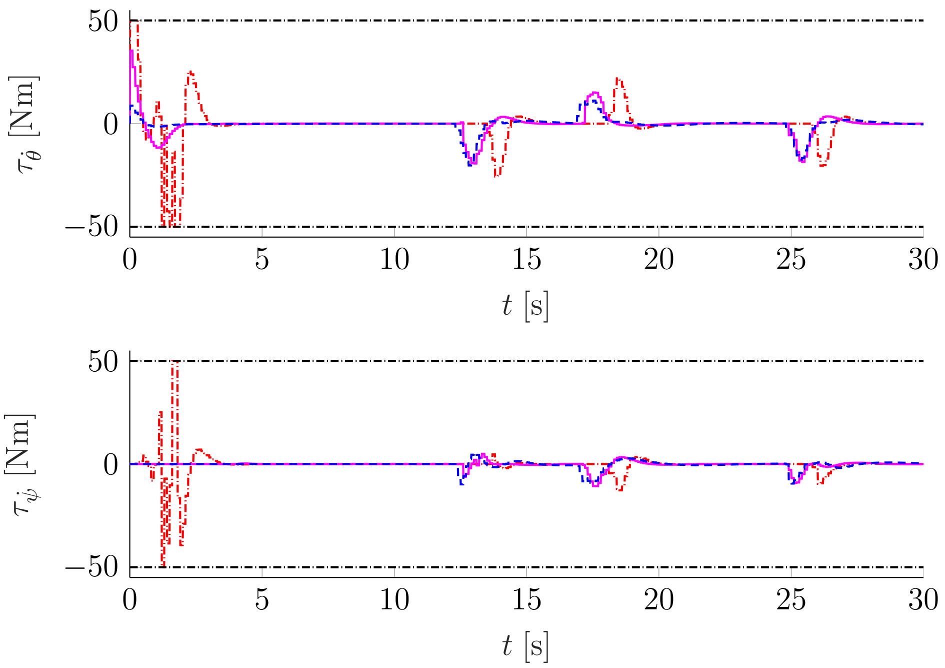

Figure 5.5: Pitch and yaw moments over simulation time applied to the AUV for the nominal case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

and pitch moments as obtained with the RHSC are given in figure 5.5. Figure, 5.6 shows the performance of the roll stabilization control loop.

The deviations from the path are smaller for the nominal case. However, as seen in figure 5.7 even for the perturbed case, the vehicle converges to the desired path as long as the tuning is not chosen to be overly aggressive. For $\rho = 0.6$, the vehicle does not reliably

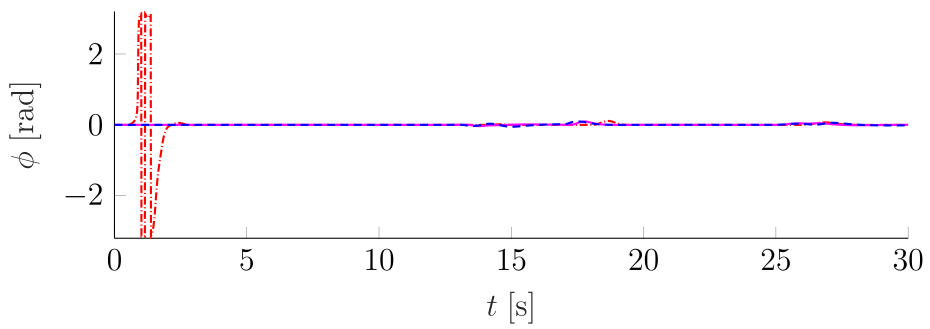

Figure 5.6: Roll angle of the AUV over simulation time for the nominal case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

converge to the desired trajectory from the initial position, indicating that the control system poorly recovers from positions far off of the desired trajectory and the vehicle does not reach segment D–E within the simulation time window. However, for $\rho = 0.975$ and $\rho = 0.9$, the tracking performance is very good and robust to noise and uncertainty. The corresponding cross-track errors are given in figure 5.8. The control signals are given in figure 5.9. It is evident that due to the noise in the position signal the actuator signals exhibit noise as well. However, tuning ρ to be larger yields less aggressive and therefore less noisy actuator commands that are still agile enough to stabilize the vehicle successfully. The roll angle is shown in figure 5.10.

The presented results were obtained on a Quadcore-CPU with 2.66 GHz and 8 GB RAM, where the simulation including the RHSC routine runs twice as fast as real-time. Hence, current resource constrained micro AUVs should possess enough computing power to run the presented RHSC.

5.5 Concluding Remarks

In summary, this chapter proposed a waypoint path following system for underactuated AUVs which is based on the receding horizon sliding control framework discussed in this dissertation. The controller combines the robustness of nonlinear feedback control methods with the advantages of MPC such as explicit handling of constraints and anticipatory control action. Simulation results show that an AUV equipped with the proposed path following system is able to follow paths with sudden directional changes in the presence of modeling uncertainties, actuator saturation, and measurement noise. The computational load remains manageable for the hardware of current micro AUVs.

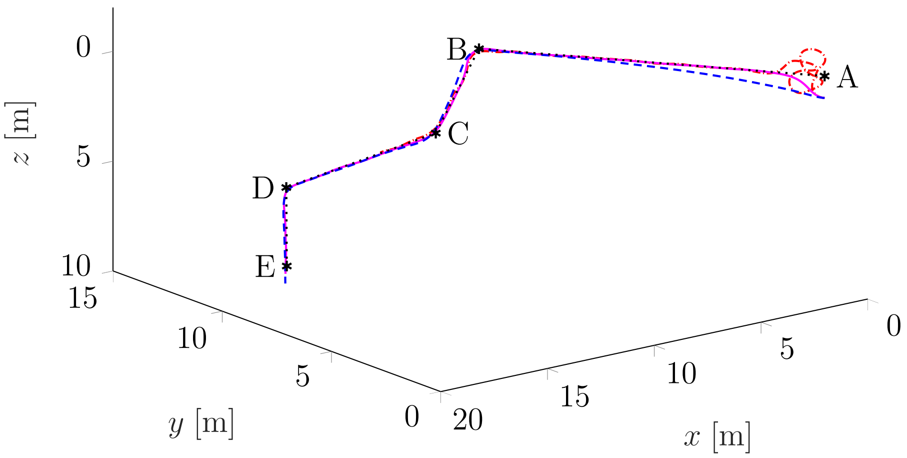

Figure 5.7: Simulated position of AUV's center of mass and the desired path (black, dotted) defined by waypoints in 3-D for perturbed case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted); the initial vehicle position at rest is $X = 1$, $Y = 0$, $Z = 0$

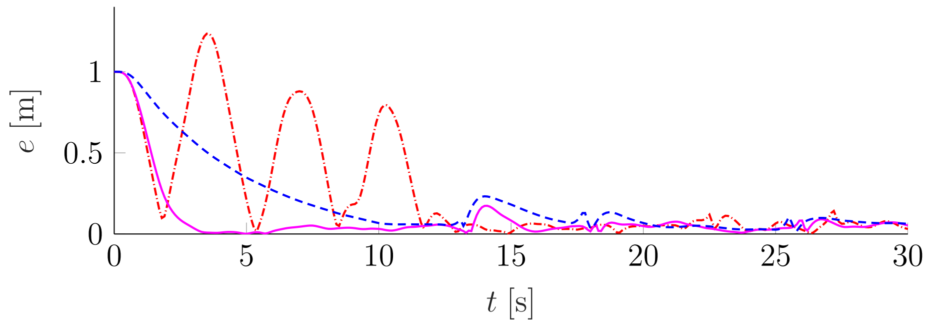

Figure 5.8: Cross-track error of the AUV over simulation time for the perturbed simulation case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

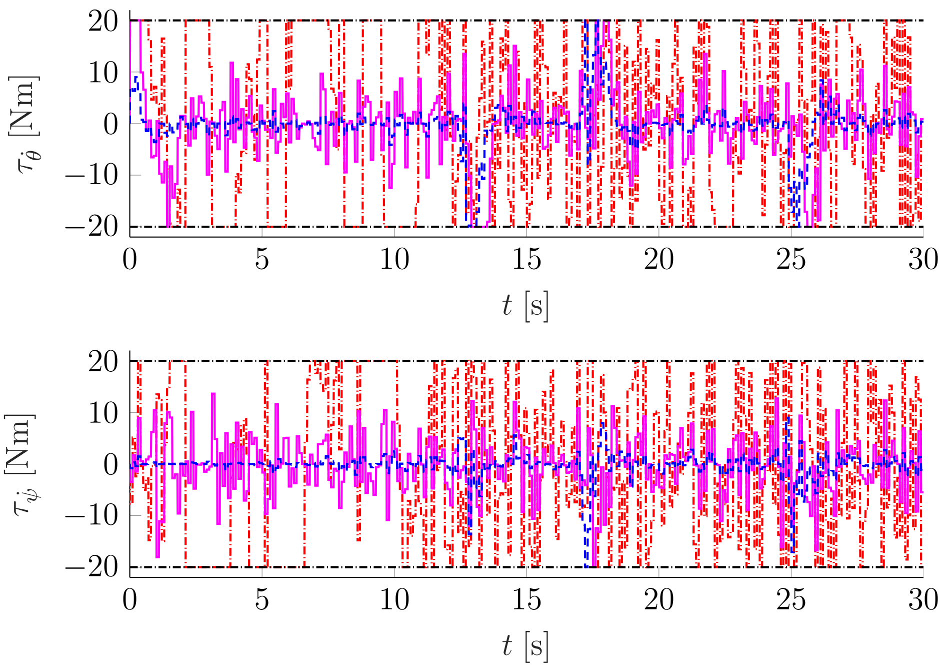

Figure 5.9: Pitch and yaw moments applied to the AUV over simulation time for the perturbed case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

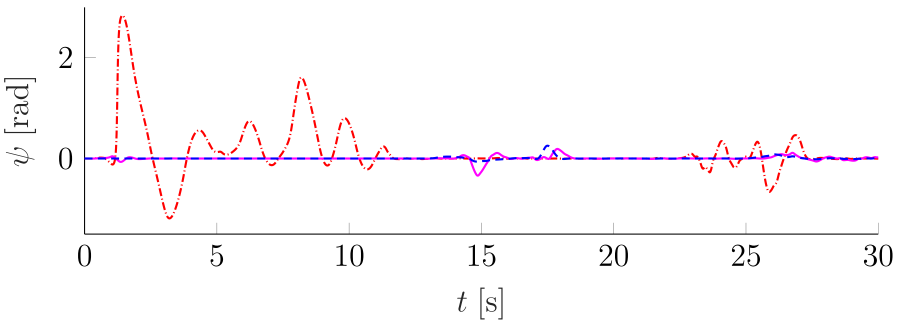

Figure 5.10: Roll angle of the AUV over simulation time for the perturbed case using a tuning of $\rho = 0.975$ (blue, dashed), $\rho = 0.9$ (magenta, solid), and $\rho = 0.6$ (red, dash-dotted)

Chapter 6

Engine Cold-Start Control

This chapter presents a performance assessment of the receding horizon sliding control algorithm. The RHSC strategy is evaluated using a classical model predictive control approach as a benchmark. The focus of this investigation is on comparing performance in terms of computational characteristics and output-tracking between the two control strategies. The case study for evaluation is a nonlinear automotive engine cold-start emissions system. It is shown that RHSC can improve the level of tracking accuracy when compared to classical MPC. At the same time, it is on average more computationally efficient for the particular application discussed in this chapter. Beyond this specific case study, the results imply that the RHSC cost function does not generally affect performance negatively compared to more established methods like MPC. Hence, RHSC can be a viable alternative to classical MPC for practical control systems. This chapter has been published in slightly modified form in [94].

6.1 Motivation

Control systems in the automotive powertrain field are becoming more and more complex and feature increasingly advanced functionality [79]. At the same time, regulations on vehicular emissions and engine safety are growing stricter annually [44]. From a control system design perspective, this motivates employing powerful control methodologies that are general enough to handle nonlinear engine dynamics, assure constraint satisfaction and yield good performance in the presence of uncertainty and noise. Receding horizon control is by far the most popular methodology that can address all of the above challenges. Therefore, it is currently of high interest to the automotive powertrain industry [44]. However, the extremely small sampling times required for engine operation are a main challenge. Moreover, the solution of nonlinear programming problems involved in MPC heavily relies on the performance of the employed iterative solvers, which is problematic from a verification and validation point of view [50].

The receding horizon sliding control method proposed in chapters 3 and 4 of this dissertation is a variant of classical MPC. Sliding surfaces can be viewed as a notational simplification [91]. By introducing a manifold with assured stability, higher dimensional tracking control problems are simplified to lower dimensional stabilization problems. RHSC adopts this simplification and combines it with a receding horizon approach. As explained in chapters 3 and 4, it minimizes the deviation of the system state from sliding surfaces in a receding horizon fashion, while accounting for system constraints. This chapter compares RHSC to classical MPC, i. e. a receding horizon controller with a conventional cost function. It is investigated whether the inclusion of sliding manifolds in a receding horizon control framework can help achieve higher algorithmic performance. Performance is measured in terms of output-tracking accuracy as well as solver characteristics such as the number of iterations and the computation time.

As a case study for comparing different receding horizon control strategies, automotive emission reduction shortly after igniting the engine is considered. Most automobiles powered by an internal combustion engine employ emissions mitigation using catalytic converters. Catalytic converters can operate at high efficiency within their operating temperature range. However, the particular focus on cold-start control is motivated by the fact that catalytic converters are largely ineffective during this initial phase of engine operation and the majority of harmful emissions occur here [13]. In order to minimize emissions, it is desirable to achieve the catalytic converter's working temperature rapidly without using excessive amounts of fuel [45]. Moreover, to avoid further emission aggravation, the engine control unit (ECU) has to carefully stabilize air-fuel-ratio while the engine temperature increases. Effectively, the fast dynamics, strong nonlinearities, as well as the physically motivated operating constraints in an automotive engine during the cold-start phase make this system a suitable case study for comparing receding horizon control algorithms.

The remainder of this chapter is structured as follows. Section 6.2 introduces a nonlinear MIMO discrete-time model of the dynamics of engine cold-start. Section 6.3 describes the application of RHSC and classical MPC to automotive emission reduction. Based on computer simulations of the engine control loop, section 6.4 evaluates and compares the RHSC and MPC control schemes. Concluding remarks are given in section 6.5.

6.2 Modeling

The mathematical engine model used in this chapter follows [45]. Relevant states for this application are the rotational engine speed, ω_{e}, the air mass in the intake manifold, m_{a}, the exhaust gas temperature, T_{exh}, and the mass flow rate of fuel into the cylinders, $\dot{m}_{\mathrm{f}}$. The system inputs are the ignition spark timing angle, Δ, the commanded mass flow rate of fuel, $\dot{m}_{\mathrm{fc}}$, and the mass flow rate of air through the throttle, $\dot{m}_{\mathrm{ai}}$. A schematic representation of the system is given in figure 6.1.

The rate of change of intake manifold air is defined by the mass flow rate of air entering the intake through the throttle, $\dot{m}_{\mathrm{ai}}$, and the air exiting the manifold into the cylinders, $\dot{m}_{\mathrm{ao}}$.

Conservation of mass yields

$$\dot{m}_{\mathrm{a}} = \dot{m}_{\mathrm{ai}} - \dot{m}_{\mathrm{ao}}(m_{\mathrm{a}}, \omega_{\mathrm{e}}). \tag{6.1}$$

The relation describing $\dot{m}_{\mathrm{ao}}$ is given by

$$\dot{m}_{\mathrm{ao}} = 0.0254 m_{\mathrm{a}} \omega_{\mathrm{e}} \eta_{\mathrm{vol}}, \tag{6.2}$$

where

$$\begin{aligned} \eta_{\mathrm{vol}} = {} & m_{\mathrm{a}}^2(-0.1636\omega_{\mathrm{e}}^2 - 7.093\omega_{\mathrm{e}} - 1750) + m_{\mathrm{a}}(0.0029\omega_{\mathrm{e}}^2 - 0.4033\omega_{\mathrm{e}} + 85.38) \\ & - (1.06 \times 10^{-5}\omega_{\mathrm{e}}^2 - 0.0021\omega_{\mathrm{e}} - 0.2719). \end{aligned} \tag{6.3}$$

The engine's rotational acceleration is set by its moment of inertia, $J = 0.1454$, and the engine torque, T_{e}. Engine torque is obtained by fitting a polynomial approximation function of engine speed and intake air mass to a torque map. See (6.4) for balance of angular momentum,

$$\dot{\omega}_{\mathrm{e}} = \frac{1}{J} T_{\mathrm{e}}(m_{\mathrm{a}}, \omega_{\mathrm{e}}). \tag{6.4}$$

The rate of change of the fuel as it evaporates in the cylinders is modeled with a first order lag and evaporation time constant $\tau_{\mathrm{f}} = 0.06$,

$$\ddot{m}_{\mathrm{f}} = \frac{1}{\tau_{\mathrm{f}}}(\dot{m}_{\mathrm{fc}} - \dot{m}_{\mathrm{f}}). \tag{6.5}$$

Exhaust gas temperature is predominantly influenced by spark timing, Δ, and air-fuel ratio, AFR, which is approximated by the mass flow rate ratio of air exiting the intake manifold to fuel entering the cylinder,

$$AFR = \frac{\dot{m}_{\mathrm{ao}}}{\dot{m}_{\mathrm{f}}}. \tag{6.6}$$

The functions SI and AI read

$$SI = 7.5\Delta + 600, \tag{6.7}$$

$$AI = 0.13\left(\cos\left(0.13\left(AFR - 13.5\right)\right)\right) \tag{6.8}$$

and define the impact of Δ and AFR on exhaust temperature, respectively. In (6.9), τ_{e} incorporates the time delay between ignition and valve opening,

$$\dot{T}_{\mathrm{exh}} = \frac{1}{\tau_{\mathrm{e}}(\omega_{\mathrm{e}})}(SI(\Delta)AI(m_{\mathrm{a}}, \omega_{\mathrm{e}}, \dot{m}_{\mathrm{f}}) - T_{\mathrm{exh}}). \tag{6.9}$$

Desired trajectories similar to those in [38] and [19] are used for ω_{e}, AFR, and T_{exh} as they were found to minimize hydrocarbon emissions. The target is to track AFR to a desired trajectory, but $\dot{m}_{\mathrm{f}}$ is easier to control, so a trajectory for $\dot{m}_{\mathrm{f}}^{\mathrm{des}} = \dot{m}_{\mathrm{ao}}/AFR^{\mathrm{des}}$ is derived from the desired air-fuel ratio.

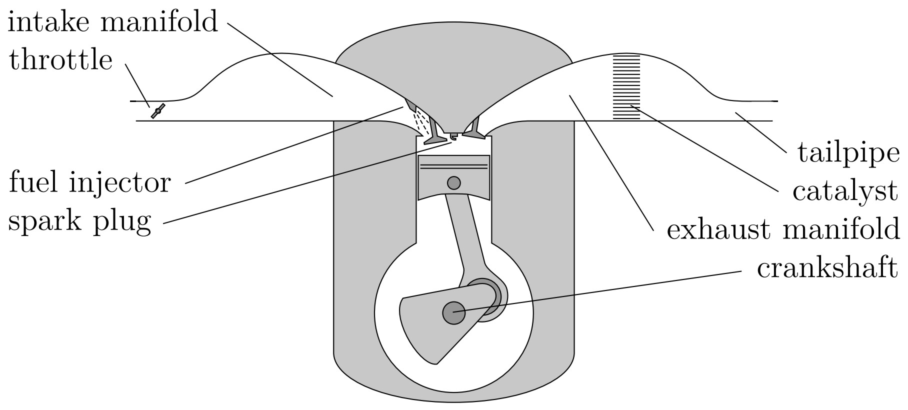

Figure 6.1: Schematic of internal combustion engine adapted from [85]

The input, state, and output vectors of the engine model at time-step k are represented by

$$\boldsymbol{x}_k = \begin{bmatrix} m_{\mathrm{a},k} & \omega_{\mathrm{e},k} & \dot{m}_{\mathrm{f},k} & T_{\mathrm{exh},k} \end{bmatrix}^\top, \tag{6.10}$$

$$\boldsymbol{u}_k = \begin{bmatrix} \dot{m}_{\mathrm{ai},k} & \dot{m}_{\mathrm{fc},k} & \Delta_k \end{bmatrix}^\top, \tag{6.11}$$

$$\boldsymbol{y}_k = \begin{bmatrix} \omega_{\mathrm{e},k} & \dot{m}_{\mathrm{f},k} & T_{\mathrm{exh},k} \end{bmatrix}^\top. \tag{6.12}$$

The control methodology developed in this dissertation requires discrete-time plant models. A forward Euler discretization scheme with sampling time T_s is applied to (6.1), (6.4), (6.5), and (6.9) in order to find discrete-time state equations of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{6.13}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k), \tag{6.14}$$

compare [75]. The measurement function $\boldsymbol{h}$ is linear and implied by (6.12).

The input constraints provide safety bounds based on physical restrictions such as ignition timing and opening time and size of the fuel and air intake valves. These lower and upper bounds on the control input are denoted by

$$\boldsymbol{u}_\mathrm{l} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_\mathrm{u}. \tag{6.15}$$

The following slew rate constraints in (6.16) were placed on the system to avoid overly rapid changes in the control signals. Similar to the input constraints, these rate bounds are evaluated at each time-step. Note that $\boldsymbol{u}_{k-1}$ is the control applied at the previous time-step,

$$-\boldsymbol{u}_\mathrm{r} \leq \boldsymbol{u}_k - \boldsymbol{u}_{k-1} \leq \boldsymbol{u}_\mathrm{r}. \tag{6.16}$$

The exact values for the input bounds are

$$\boldsymbol{u}_{\mathrm{l}} = \begin{bmatrix} 0 \\ 0 \\ -10 \end{bmatrix} \quad \boldsymbol{u}_{\mathrm{u}} = \begin{bmatrix} 0.0063 \\ 0.0041 \\ 40 \end{bmatrix} \quad \boldsymbol{u}_{\mathrm{r}} = \begin{bmatrix} 0.012 \\ 0.04099 \\ 5 \end{bmatrix} \tag{6.17}$$

.

6.3 Control System Design

This section presents the application of the RHSC and classical MPC methodologies to the problem of automotive cold-start emission control.

6.3.1 RHSC Design

With the cold-start model defined, a specific RHSC control strategy can now be formulated. First, specify an error vector, $\boldsymbol{e}_k$, as

$$\boldsymbol{e}_k = \begin{bmatrix} \omega_{\mathrm{e},k} - \omega_{\mathrm{e},k}^{\mathrm{des}} & \dot{m}_{\mathrm{f},k} - \dot{m}_{\mathrm{f},k}^{\mathrm{des}} & T_{\mathrm{exh},k} - T_{\mathrm{exh},k}^{\mathrm{des}} \end{bmatrix}^\top . \tag{6.18}$$

Let a sliding variable vector be defined as

$$\boldsymbol{s}_k = \begin{bmatrix} \rho e_{1,k} - e_{1,k+1} & e_{2,k} & e_{3,k} \end{bmatrix}^\top . \tag{6.19}$$

Note that the outputs T_{exh} and $\dot{m}_{\mathrm{f}}$ have relative degree equal to one, whereas ω_{e} has a relative degree of two. Therefore, the engine speed output error dynamics require a first order difference operator to ensure that $s_{1,k+1}$ is an explicit function of a control input. The tuning parameter is restricted to the range $0 < \rho < 1$.

As usual, define the shorthand notation

$$\boldsymbol{S}_{k+1} = \begin{bmatrix} \boldsymbol{s}_{k+1} & \dots & \boldsymbol{s}_{k+N} \end{bmatrix} \tag{6.20}$$

and let

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N+1} \end{bmatrix} , \tag{6.21}$$

$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}_k & \dots & \boldsymbol{u}_{k+N} \end{bmatrix} . \tag{6.22}$$

A suitable CFTOC problem for the seeked cold-start RHSC can then be formulated as

$$\begin{aligned}
\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \quad & \|\boldsymbol{W}\boldsymbol{S}_{k+1}\|_{\mathrm{F}}^2 && (6.23)\\
\text{s.t.} \quad & \boldsymbol{s}_i = \mathscr{D}(\boldsymbol{e}_i),\ i = k+1,\dots,k+N \\
& \boldsymbol{e}_i = \boldsymbol{h}(\boldsymbol{x}_i) - \boldsymbol{y}_i^{\mathrm{des}},\ i = k+1,\dots,k+N+1 \\
& \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i,\boldsymbol{u}_i),\ i = k,\dots,k+N \\
& -\boldsymbol{u}_{\mathrm{r}} \le \boldsymbol{u}_i - \boldsymbol{u}_{i-1} \le \boldsymbol{u}_{\mathrm{r}},\ ,i = k,\dots,k+N \\
& \boldsymbol{u}_{\mathrm{l}} \le \boldsymbol{u}_i \le \boldsymbol{u}_{\mathrm{u}},\ i = k,\dots,k+N \\
& \boldsymbol{x}_k = \boldsymbol{x}(k) && (6.24)\\
& \boldsymbol{u}_{k-1} = \boldsymbol{u}(k-1),
\end{aligned}$$

where an additional diagonal weighting matrix, $\boldsymbol{W} \in \mathbb{R}^{3\times 3}$, was introduced. Hence, the total number of tuning parameters specific to this control scheme is 4.

6.3.2 Classical MPC Design

As a benchmark, the delta input formulation introduced in section 2.4 is applied to the cold-start problem. A classical choice for the MPC objective function is used, where input changes and tracking errors are penalized with quadratic functionals. The resulting CFTOC problem reads as follows,

$$\begin{aligned}
\min_{\boldsymbol{X}_k,\boldsymbol{U}_k} \quad & \|\boldsymbol{y}_{k+N} - \boldsymbol{y}_{k+N}^{\mathrm{des}}\|_{\boldsymbol{Q}}^2 + \sum_{i=0}^{N-1} \|\boldsymbol{y}_{k+i} - \boldsymbol{y}_{k+i}^{\mathrm{des}}\|_{\boldsymbol{Q}}^2 + \|\boldsymbol{u}_{k+i} - \boldsymbol{u}_{k+i-1}\|_{\boldsymbol{R}}^2 && (6.25)\\
\text{s.t.} \quad & \boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i,\boldsymbol{u}_i),\ i = k,\dots,k+N-1 \\
& \boldsymbol{y}_i = \boldsymbol{h}(\boldsymbol{x}_i,\boldsymbol{u}_i),\ i = k,\dots,k+N \\
& -\boldsymbol{u}_{\mathrm{r}} \le \boldsymbol{u}_i - \boldsymbol{u}_{i-1} \le \boldsymbol{u}_{\mathrm{r}},\ ,i = k,\dots,k+N-1 \\
& \boldsymbol{u}_{\mathrm{l}} \le \boldsymbol{u}_i \le \boldsymbol{u}_{\mathrm{u}},\ i = k,\dots,k+N-1 \\
& \boldsymbol{x}_k = \boldsymbol{x}(k) \\
& \boldsymbol{u}_{k-1} = \boldsymbol{u}(k-1).
\end{aligned}$$

Now, $\boldsymbol{Q} \in \mathbb{R}^{3\times 3}$ and $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ in (6.25) can be selected in order to prioritize output tracking and incorporate input penalties. This study is restricted to the standard case where $\boldsymbol{Q}$ and $\boldsymbol{R}$ are diagonal matrices leading to 6 tuning parameters specific to this MPC scheme.

6.4 Results

Implementation details and simulation results related to the above controller synthesis' are presented and evaluated in the following.

6.4.1 Implementation

Both closed-loop systems were simulated in MATLAB/SIMULINK using C code based S-functions for implementing the two considered predictive control schemes. In particular, the algorithms used the solver NPSOL [34] for the discrete-time controllers with a sampling time of $T_s = 16\,\text{ms}$. Likewise, the discrete-time plant dynamics were integrated with fixed integration time-steps of 16 ms. A prediction horizon length of $N = 4$ steps was used in both control strategies.

6.4.2 Tuning

Tuning substantially affects algorithmic performance. To obtain a fair comparison, the two considered control algorithms were tuned using a joint performance index that considers tracking performance and computational characteristics. A numerical routine was used to obtain optimal tuning for both controllers with respect to this joint performance metric. The tuning parameter ρ and the weights $\boldsymbol{W}$, $\boldsymbol{Q}$, and $\boldsymbol{R}$ for the respective approaches were determined with the aid of a genetic algorithm solver.

6.4.3 Numerical Simulations

Both closed-loop systems were simulated using the aforementioned desired output trajectories to produce the tracking performance seen in figure 6.2. The figure shows the tracking responses of all three outputs for the RHSC and MPC schemes. All three outputs contain initial tracking inaccuracies that are more pronounced in the MPC case, especially exhaust temperature and air-fuel ratio. For engine speed, the receding horizon sliding control method appears to eliminate the initial decaying oscillations visible in the MPC approach. For air-fuel ratio the RHSC approach avoids a large overshoot initially present in the MPC case. The exhaust temperature also rises to the desired value of 650°C faster with the RHSC method than with the classical MPC strategy.

The comparative error tracking performances are also reflected in table 6.4.3. The tracking error statistics show a clear and promising decrease in average and cumulative errors for engine speed, air-fuel ratio, and exhaust temperature tracking. The tracking errors are also significantly more consistent for all three outputs as seen by the standard deviations for MPC and RHSC. The exhaust temperature tracking error in RHSC outperforms that of MPC in average error, standard deviation, and cumulative error in spite of an initial overshoot.

Figure 6.3 shows the simulated control inputs for both control strategies. All control inputs for each case follow a smooth response, but the inputs of $\dot{m}_{\text{ai}}$ and $\dot{m}_{\text{fc}}$ in the MPC case oscillate initially. The $\dot{m}_{\text{ai}}$ signal from the MPC reaches saturation during these oscillations whereas the respective RHSC input changes less abruptly. For spark timing, the signal from RHSC briefly oscillates well below saturation but then smoothly approaches a steady value.

The computational performance of both controllers is displayed in figure 6.4. The solver flag status indicates a diagnostic result after the NPSOL solver has been called. A flag of

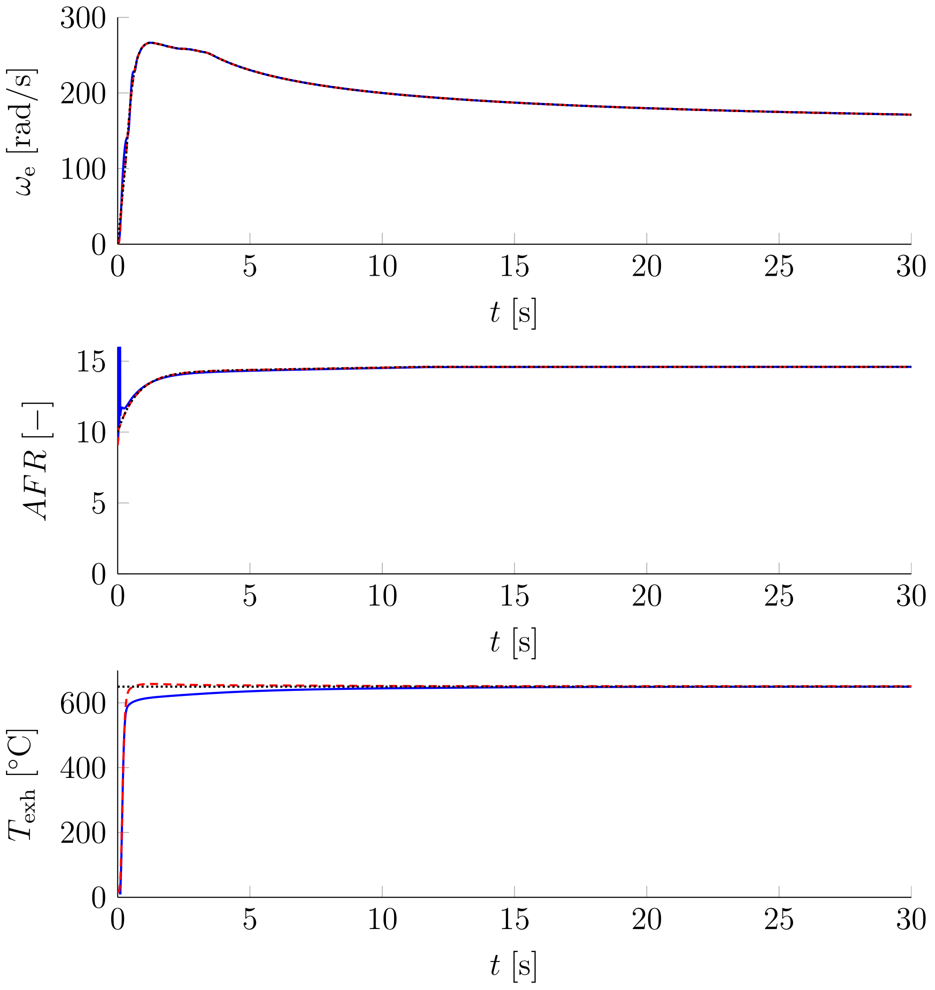

Figure 6.2: Reference signals (black, dotted) and engine outputs from closed-loop simulations using MPC (blue, solid) and RHSC (red, dashed) over simulation time

0 indicates no warnings from the solver. The only nonzero flags during the RHSC-based simulation occur during the beginning of the simulation and are all 4, indicating a jump to the maximum number of solver iterations at that time-step [34]. The sequential quadratic programming (SQP) iterations plot for both control methods shows RHSC predominantly requires less iterations, except for the infrequent case where the iteration limit is reached. Similarly for NPSOL CPU time, the RHSC solver time is shorter on average compared to MPC solver time.

These aforementioned general trends are also reflected in table 6.4.3. The table shows a clear decrease in average and cumulative solver times and iterations with RHSC. The RHSC method uses 33.333% less CPU time for NPSOL and solver iterations are decreased by 32.093% when compared to MPC.

Table 6.1: Comparison of the mean absolute tracking errors, the corresponding standard deviations, and the cumulative error values for engine speed, air-fuel ratio, and exhaust temperature for the MPC and RHSC schemes

Error Signal	Controller	Average Absolute Error	Standard Deviation	Cumulative Error
Engine Speed	MPC	0.2871	2.3171	538.57
	RHSC	0.0765	0.8850	143.46
Air-fuel Ratio	MPC	0.0641	1.1278	120.32
	RHSC	0.0008	0.0255	1.6074
Exhaust Temperature	MPC	10.971	48.527	20583
	RHSC	6.7075	46.904	12583

Table 6.2: Comparison of the mean values, the corresponding standard deviations, and the cumulative values for the NPSOL CPU time and iteration numbers for the MPC and RHSC schemes

Solver Feature	Controller	Average Value	Standard Deviation	Cumulative Value
CPU Time	MPC	0.0027	0.0001	5.0765
	RHSC	0.0018	0.0003	3.4470
Iterations	MPC	25.055	1.7600	47004
	RHSC	17.014	6.1069	31920

6.5 Concluding Remarks

In this chapter, the RHSC method has been compared to a standard model predictive control approach for a practical multi-input multi-output system. The case of engine cold-start emissions has been chosen due to its nonlinearity, fast dynamics, and physical constraints. Evidently, the novel method improves tracking for all three outputs while requiring less tuning parameters, computation time and solver iterations. To conclude, depending on the control system at hand, RHSC can be a viable alternative to MPC that potentially yields better tracking performance without compromising computational efficiency.

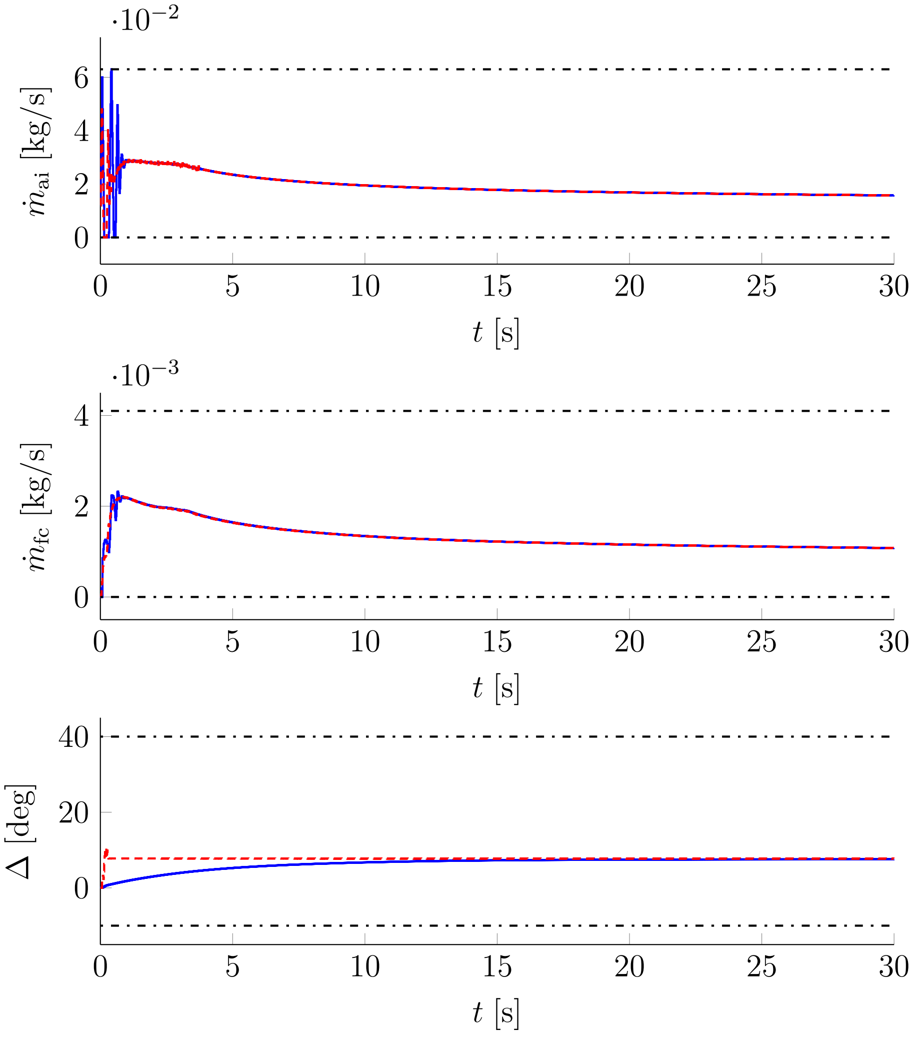

Figure 6.3: Control inputs from closed-loop simulations using MPC (blue, solid) and RHSC (red, dashed) and corresponding upper and lower bounds (black, dash-dotted) over simulation time

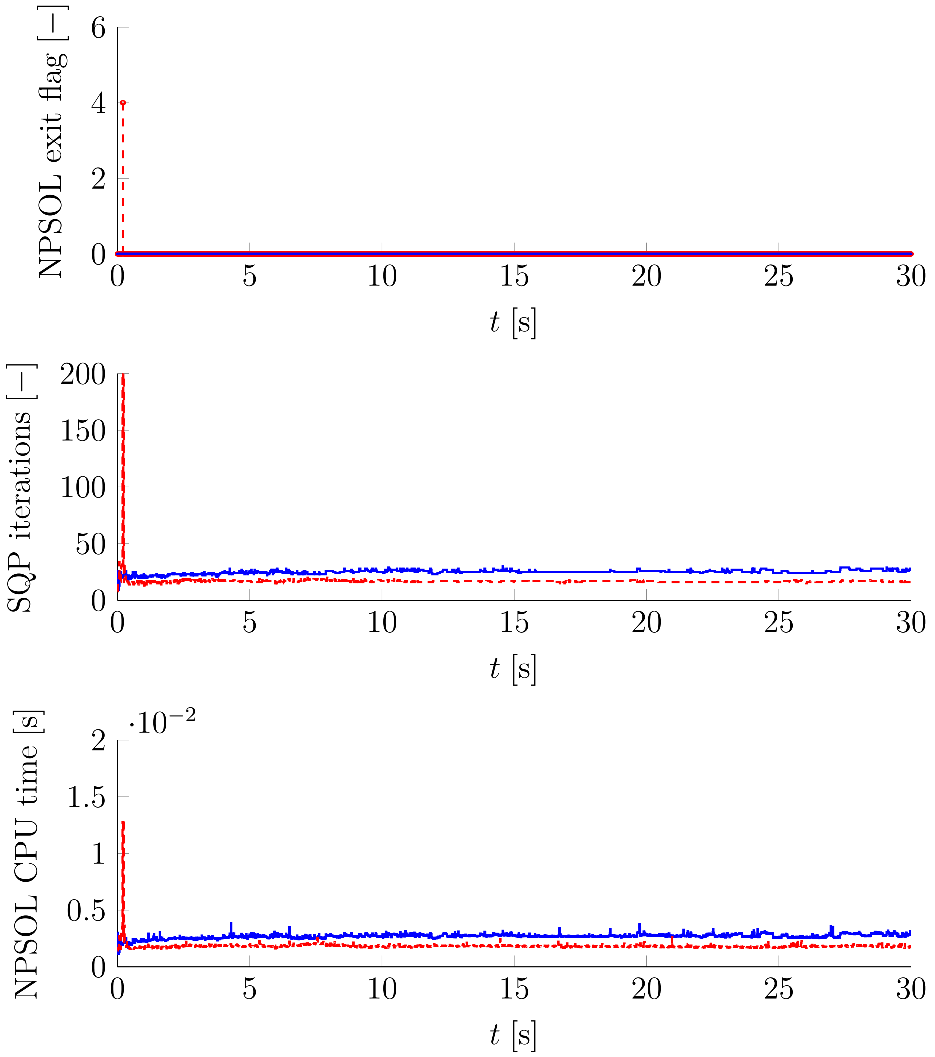

Figure 6.4: Solver exit flags, required number of NPSOL SQP iterations, and CPU execution time for the MPC (blue, solid) and RHSC (red, dashed)

Chapter 7

Severe Autonomous Driving Maneuvers

This chapter implements RHSC on a real-world test vehicle for autonomously performing the severe obstacle avoidance double lane change according to the ISO 3888-2 standard. The vehicle is equipped with light detection and ranging (LIDAR) sensors as well as a global positioning system (GPS) and an inertial measurement unit (IMU). The RHSC algorithm is complemented with an unscented Kalman filter (UKF) for estimating the vehicle state and performing simultaneous localization and mapping (SLAM) of the environment online based on the sensor data. Since RHSC inherits simple tuning rules from sliding control, the tuning effort of the resulting controller developed in this chapter is minimal. A sequential quadratic programming solver is utilized for repeatedly solving the nonlinear program arising from the RHSC design. This case study confirms the effectiveness, practicality, and real-time capabilities of RHSC with simulations and full-scale experiments.

7.1 Motivation

As summarized in chapter 3, combining model predictive control and sliding mode control in one control scheme has attracted the interest of several researchers. Different approaches for merging both methodologies have been proposed in the automatic control literature and were outlined in chapter 3, e. g. [102, 101, 98, 78, 65, 64]. Furthermore, the application of hybrid MPC/SMC methodologies to physical systems was previously studied in the literature referenced in chapter 3. For instance, in [16] generalized predictive control is combined with sliding mode control for linear plants and the application of a chemical process is considered in simulation trials. The works [76, 31] present hybrid MPC/SMC schemes for linear plus dead time systems as well as experimental results of their application to a distributed solar collector field and a solar air conditioning plant, respectively. In both experiments the sampling intervals were several seconds long.

Along a similar line of thought, the receding horizon sliding control technique has been proposed in this dissertation. The developments in chapters 3 and 4 show that RHSC translates an output tracking objective into a desired state space objective by designing sliding surfaces. Sliding surfaces directly encode smooth error dynamics and the control is used to minimize the distance of the system state to these manifolds. The performance of RHSC was compared against classical MPC in chapter 6 and [94], respectively, for an automotive combustion engine application and in [63] for an electric vehicle powertrain transmission. It was shown that RHSC can yield superior performance while using less computational effort for these specific applications. Also, in [63] it is stated that the tuning of RHSC appears to be simplified versus classical MPC due to the intuitive interpretation of the tuning parameters.

Experimental testing of RHSC is beyond the scope of the previous chapters 5, 6 and the papers [39, 40, 63, 94]. Therefore, it is yet to be shown that RHSC yields acceptable performance in real-world experiments. In fact, to the author's knowledge there is no study where hybrid MPC/SMC schemes were applied to high speed systems with sampling times of the order of fractions of seconds. In order to fill this gap, this chapter implements RHSC within an observer-based control loop for the fast dynamics of autonomous vehicle handling. As a test case, the ISO 3888-2 double lane change is utilized [48]. Traditionally, this ISO standard is used to evaluate the handling capabilities of human-steered passenger cars. A severe obstacle avoidance course is set up with cones and a professional test driver performs the test at incrementally higher speeds up to a point where it cannot be completed without failure anymore. While it is current engineering practice to evaluate a vehicle's handling performance with a human test driver, using a control system yields far more objective and reproducible results on the ISO 3888-2 test. For this purpose, for example the company VEHICO currently offers commercial steering robots for standardized vehicle tests.

Farther, research in the area of autonomous vehicles is motivated by the enormous potential of such systems to increase traffic safety and efficiency, hence, autonomous driving is considered one of the high-impact research challenges in systems and control at this time [57]. More specifically, predictive control is of interest because it can compensate for feasibility deficiencies of motion plans generated higher up in the autonomous vehicle decision-making hierarchy [74]. In [27] it is emphasized that autonomous vehicles have to be capable of reacting intelligently and safely in emergency scenarios. Emergency scenarios can occur due to several reasons including unforeseen changes in the environment, spurious sensor readings, or temporary control system failures. Safe control during emergency scenarios may require maneuvering the vehicle at the limits of handling where nonlinear effects such as tire dynamics have a significant influence on the vehicle dynamics [30]. One way of ensuring the stability of autonomous vehicles during severe maneuvers is the classical double lane change test as utilized in [20], where a fixed reference trajectory is tracked on slippery roads. Hence, it can be concluded that automation of standardized test scenarios is not only of interest for the development of traditional human-steered vehicles but it is also essential for evaluating the agility and stability of self-driving vehicles.

In summary, using the ISO 3888-2 standard, this is the first work that validates the functionality of RHSC in real-world experiments using an observer-based control strategy that includes environment perception. The structure of this chapter is such that nonlinear vehicle and environment modeling is addressed in section 7.2. Subsequently, section 7.3 discusses the core algorithms for path following control, RHSC, as well as localization and environment mapping, UKF-SLAM, as utilized in this work. Section 7.4 describes the application of the developments from sections 7.2 and 7.3 for the specific case of the severe obstacle avoidance double lane change maneuver. Section 7.5 presents simulations as well as experimental results. Finally, conclusions are drawn in section 7.6.

7.2 Modeling

This section introduces a vehicle model, a simple model for the surrounding environment, as well as measurement models that will be utilized throughout the remainder of this chapter.

7.2.1 States and Transition Model

Figure 7.1 illustrates the considered vehicle and the environment. The inertial coordinate system is $O(X_O, Y_O, Z_O)$. Planar motion perpendicular to the Z_O axis is assumed. The velocity of the vehicle's center of gravity (COG) in inertial coordinates is described by

$$\frac{\mathrm{d}X}{\mathrm{d}t} = \dot{x}\cos(\psi) - \dot{y}\sin(\psi), \tag{7.1}$$

$$\frac{\mathrm{d}Y}{\mathrm{d}t} = \dot{x}\sin(\psi) + \dot{y}\cos(\psi), \tag{7.2}$$

where ψ is the yaw angle with respect to the inertial coordinate system. In (7.1), (7.2) the symbols $\dot{y}$ and $\dot{x}$ denote the lateral and longitudinal velocity, respectively.

The yaw, lateral, and longitudinal dynamics are modeled using a dynamic bicycle model [80, 42],

$$\frac{\mathrm{d}\psi}{\mathrm{d}t} = \dot{\psi}, \tag{7.3}$$

$$\frac{\mathrm{d}\dot{\psi}}{\mathrm{d}t} = \frac{1}{I_z}\left(\cos(\delta) l_f F_{yf} - l_r F_{yr}\right), \tag{7.4}$$

$$\frac{\mathrm{d}\dot{x}}{\mathrm{d}t} = \frac{1}{M}\left(F_x - \sin(\delta) F_{yf}\right) + \dot{y}\dot{\psi} + a, \tag{7.5}$$

$$\frac{\mathrm{d}\dot{y}}{\mathrm{d}t} = \frac{1}{M}\left(\cos(\delta) F_{yf} + F_{yr}\right) - \dot{x}\dot{\psi}. \tag{7.6}$$

The vehicle mass is symbolized by M and I_z is the rotational inertia around the yaw axis. The lengths l_f and l_r denote the longitudinal distance of the front and rear axles from the vehicle's COG. The lateral acceleration is $\ddot{y}$ and the yaw rate and acceleration are $\dot{\psi}$ and $\ddot{\psi}$,

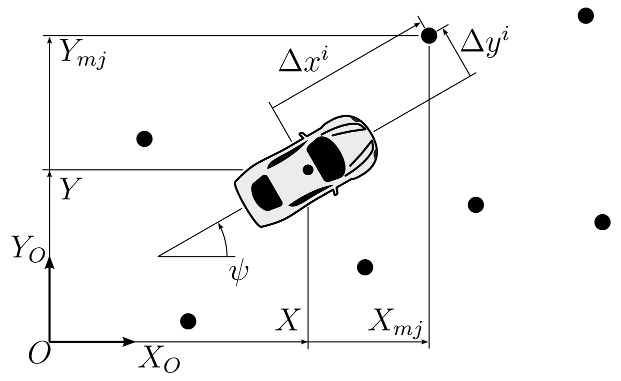

Figure 7.1: Illustration of geometric quantities involved in modeling the considered vehicle control scenarios

respectively. The steering angle is represented by δ and the vehicle acceleration input is a. The front and rear tire cornering forces, F_{yf} and F_{yr}, are modeled using the magic formula [73]

$$F_{yl} = 2D_l \sin\left(C_l \arctan\left((1-E_l)B_l\alpha_l + E_l \arctan(B_l\alpha_l)\right)\right). \tag{7.7}$$

The index l is either f or r representing the front or rear tire. Moreover, B_l, C_l, D_l, and E_l are empirical constants that characterize the respective tire. Finally, the relations

$$\alpha_f = \delta - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}, \tag{7.8}$$

$$\alpha_r = -\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}, \tag{7.9}$$

determine the tire slip angles in (7.7). Finally, the longitudinal force F_x is modeled with the equation

$$F_x = -\frac{1}{2}d_a C_d A_f \dot{x}^2 - R_x, \tag{7.10}$$

where d_a is the air density, C_d is the drag coefficient, A_f is the frontal area of the vehicle, and R_x denotes the vehicle's rolling resistance [80].

Applying Euler discretization, i. e. approximating time-derivatives by forward finite differences, equations (7.1), (7.2), (7.3), (7.4), (7.5), (7.6) can be transformed into a discrete-time state space model of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{\theta}_k, \tag{7.11}$$

where the state vector of dimension $n = 6$ is

$$\boldsymbol{x}_k = \begin{bmatrix} X_k & Y_k & \psi_k & \dot{\psi}_k & \dot{x}_k & \dot{y}_k \end{bmatrix}^\top \tag{7.12}$$

and the control vector is

$$\boldsymbol{u}_k = \begin{bmatrix} a_k & \delta_k \end{bmatrix}^\top. \tag{7.13}$$

In (7.11) the discrete time-step is indicated by the subscript k or $k+1$, respectively, and the sampling time is T_s. An n component zero-mean white Gaussian noise term $\boldsymbol{\theta}_k$ with covariance $\boldsymbol{\Theta}_k$ was added to the state equation to represent system uncertainty.

The environment is modeled with n_m point landmarks. The environment model or map is denoted by

$$\boldsymbol{m}_k = \begin{bmatrix} X_{m1,k} & Y_{m1,k} & \dots & X_{mn_m,k} & Y_{mn_m,k} \end{bmatrix}^\top, \tag{7.14}$$

where the $X_{mj,k}$ and $Y_{mj,k}$ denote the locations of landmark j at time k with respect to $O(X_O, Y_O, Z_O)$. Assume there is an environment motion model of the form $\boldsymbol{m}_{k+1} = \boldsymbol{p}(\boldsymbol{m}_k) + \boldsymbol{\xi}_k$ available, where $\boldsymbol{\xi}_k$ is white Gaussian noise with covariance $\boldsymbol{\Xi}_k$. If the environment is static as in this work, this model reduces to $\boldsymbol{m}_{k+1} = \boldsymbol{m}_k$. A detailed discussion of environment prediction is beyond the scope of this work, for more details see e. g. [21] or the motion prediction survey [58].

Finally, similar to [95], define the overall state by concatenating the vehicle and environment states as $\bar{\boldsymbol{x}}_k = \begin{bmatrix} \boldsymbol{x}_k^\top & \boldsymbol{m}_k^\top \end{bmatrix}^\top$. The length of the overall state vector is $\bar{n} = n + 2n_m$. Note that the dynamics of the environment and vehicle are modeled under the approximate assumption that both evolve independently.

7.2.2 Observation Models

The vehicle measurement or observation model relates the vehicle state $\boldsymbol{x}_k$ to a measurement $\boldsymbol{y}_k$. It can be formulated as

$$\boldsymbol{y}_k = \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{\phi}_k. \tag{7.15}$$

A zero mean white Gaussian noise term, $\boldsymbol{\phi}_k$, with covariance matrix $\boldsymbol{\Phi}_k$ is added to model sensor noise. In the ideal case where all state variables can be measured directly, the measurement function simplifies to the linear relation $\boldsymbol{g}(\boldsymbol{x}_k) = \boldsymbol{x}_k$.

Furthermore, it is assumed that the vehicle is equipped with sensory equipment to perceive the environment. In particular, the relative distances Δx, Δy of landmarks/features are assumed to be measurable. Assuming there is a large number of those measurements, they are indexed by i. By indexing the features by j, the measurement function can be easily written as the following geometric relation

$$\begin{bmatrix} \Delta x_k^i \\ \Delta y_k^i \end{bmatrix} = \begin{bmatrix} \cos(\psi_k) & \sin(\psi_k) \\ -\sin(\psi_k) & \cos(\psi_k) \end{bmatrix} \begin{bmatrix} X_{mj,k} - X_k \\ Y_{mj,k} - Y_k \end{bmatrix}. \tag{7.16}$$

By letting $\boldsymbol{z}_k^i = \begin{bmatrix} \Delta x_k^i & \Delta y_k^i \end{bmatrix}^\top$, equation (7.16) can be compactly rewritten as

$$\boldsymbol{z}_k^i = \boldsymbol{h}^j(\bar{\boldsymbol{x}}_k) + \boldsymbol{\omega}_k, \tag{7.17}$$

representing a measurement model with an added zero mean white Gaussian noise term, $\boldsymbol{\omega}_k$, with covariance $\boldsymbol{\Omega}_k$ depending on the vehicle sensor equipment. It should be noted that equations (7.16) and (7.17) assume that the correspondence of measurement i to landmark j is known. However, in practice for a given measurement it is a priori unknown which object it corresponds to. This correspondence issue will be resolved in section 7.3 as part of the UKF-SLAM algorithm.

7.3 Algorithm Design

This section discusses path following control and perception. The presented RHSC and UKF-SLAM algorithms constitute the core of the autonomous vehicle control system that will be further detailed in section 7.4.

7.3.1 RHSC for Path Following

The focus of this section is on deriving a path following controller by utilizing the RHSC design approach proposed in this dissertation. To recapitulate, the main design concept in RHSC are sliding surfaces, which are manifolds in state space where predefined desired error dynamics hold. The RHSC design phase and tuning is quite similar to the classical discrete-time sliding control technique. This chapter exploits the fact that the combination of sliding control concepts and receding horizon control enables the controller to account for constraints and to effectively use future reference information, while the design and tuning remain similar to discrete sliding control.

The main control actuator for path following control is the steering wheel. Since longitudinal velocity control is secondary, for the remainder of this section it is assumed that there is an explicit state feedback law of the form $a = c(\boldsymbol{x}_k)$ available for speed control, this control law will be further detailed in section 7.4.3. Furthermore, assume the vehicle is traveling primarily in the X_O direction. On curved roads, this may require reorientation of the coordinate system $O(X_O, Y_O, Z_O)$ at successive time-steps depending on the current orientation of the road. In practice, for instance lane markers can be utilized for properly orienting coordinates.

With the above assumptions in place, sliding surface design is the next step and the development in this chapter is similar to [39, 14]. Figure 7.2 illustrates the path following problem. First, define the desired trajectory in terms of desired Y_O-location and heading as

$$\boldsymbol{r}_k = \begin{bmatrix} Y_k^{\mathrm{des}} & \psi_k^{\mathrm{des}} \end{bmatrix}^\top . \tag{7.18}$$

Next, using the shorthand notation $b(\boldsymbol{x}_k, \boldsymbol{r}_k)$, define an error as a weighted sum of the position errors, $e_{Y,k} = Y_k - Y_k^{\mathrm{des}}$, and the yaw angle error, $e_{\psi,k} = \psi_k - \psi_k^{\mathrm{des}}$, as

$$e_k = e_{Y,k} + \eta e_{\psi,k} =: b(\boldsymbol{x}_k, \boldsymbol{r}_k). \tag{7.19}$$

The weighting parameter $\eta > 0$ is used for tuning and it determines the relative weight on the position and yaw error.

Second, following [47, 89, 39] a sliding variable, s_k, with relative degree [55] equal to one is defined using a stable difference operator on the tracking error. The difference operator

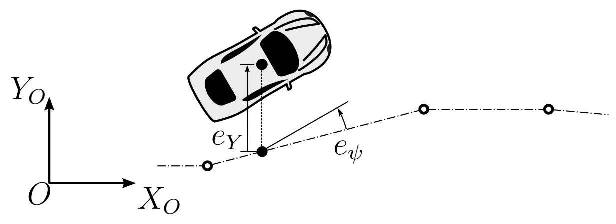

Figure 7.2: Illustration of the vehicle path following problem as well as the position and yaw errors

is denoted by the shorthand notation $\mathscr{D}(e)_k$. Like many mechanical systems, the problem at hand has a relative degree of two. Hence, a first order difference operator is defined as required in chapter 3,

$$s_k = \rho e_k - e_{k+1} =: \mathscr{D}(e)_k. \tag{7.20}$$

The tuning parameter ρ determines the desired error convergence rate on the sliding surface and a reasonable range is $0 < \rho < 1$. Note that stability of the difference operator ensures that $s_k \equiv 0$ implies $e_k \to 0$ as $k \to \infty$, i. e. driving the sliding variable to zero yields asymptotic error convergence to zero. Moreover, as in classical discrete sliding control, s_{k+1} is an explicit function of the control δ_k after substituting the system equations recursively [40].

The goal of RHSC is to drive the system to the possibly nonlinear and time-varying sliding surface by using a receding horizon approach. Therefore, define a vector that contains the s quantity over a receding prediction horizon that reaches N steps into the future,

$$\boldsymbol{S}_{k+1} = \begin{bmatrix} s_{k+1} & \dots & s_{k+N} \end{bmatrix}. \tag{7.21}$$

Similarly, the following shorthand notation is used to represent the sequences of future states, steering controls, and reference values

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k & \dots & \boldsymbol{x}_{k+N+1} \end{bmatrix}, \tag{7.22}$$

$$\boldsymbol{\Delta}_k = \begin{bmatrix} \delta_k & \dots & \delta_{k+N} \end{bmatrix}, \tag{7.23}$$

$$\boldsymbol{R}_k = \begin{bmatrix} \boldsymbol{r}_{k+1} & \dots & \boldsymbol{r}_{k+N+1} \end{bmatrix}. \tag{7.24}$$

Then, the following constrained finite-time optimal control problem can be defined,

$$\min_{\boldsymbol{X}_k, \boldsymbol{\Delta}_k} \quad \|\boldsymbol{S}_{k+1}^{\top}\|_2^2 \tag{7.25}$$

$$\begin{aligned}
\text{s.t.} \quad & s_i = \mathscr{D}(e)_i, \ i = k+1, \dots, k+N \\
& e_i = b(\boldsymbol{x}_i, \boldsymbol{r}_i), \ i = k+1, \dots, k+N+1 \\
& \boldsymbol{x}_{i+1} = \boldsymbol{f}\left(\boldsymbol{x}_i, \begin{bmatrix} c(\boldsymbol{x}_i) & \delta_i \end{bmatrix}^{\top}\right), \ i = k, \dots, k+N \\
& -\bar{\dot{\delta}} \leq \delta_i - \delta_{i-1} \leq \bar{\dot{\delta}}, \ i = k, \dots, k+N \\
& -\bar{\delta} \leq \delta_i \leq \bar{\delta}, \ i = k, \dots, k+N \\
& \boldsymbol{x}_k = \hat{\boldsymbol{x}}_k \\
& \delta_{k-1} = \delta(k-1).
\end{aligned} \tag{7.26}$$

This optimization problem is initialized with an estimate of the vehicle state, $\hat{\boldsymbol{x}}_k$. It includes input and input rate constraints on the steering angle. The problem is solved at every time-instant, the first element of the resulting control sequence is implemented and the procedure is repeated at the next time-step with the horizon shifted by one step as usual.

7.3.2 State Estimation with UKF-SLAM

This work deploys unscented Kalman filtering [95] for simultaneous localization and mapping. UKF-SLAM was previously addressed in [70] and [86], but overall the method has gained little attention in the literature. This is surprising given that UKF is shown to behave well under some of the typical nonlinearities found in vehicle dynamics and robotics such as rotations [52]. The filter for estimating the vehicle state and the state of the environment follows the standard predictor (pred.)/corrector (corr.) scheme [95]. Algorithm 5 summarizes the individual steps of the procedure that will be further detailed in the following.

First, line 3 of algorithm 5 is discussed in detail. This UKF prediction step uses the vehicle model and the control signal to predict the value of the vehicle state at the current time-step, given information from the previous time-step. Consider the following partition of the overall mean and covariance matrix from the previous time-step

$$\hat{\bar{\boldsymbol{x}}}_{k-1} = \begin{bmatrix} \hat{\boldsymbol{x}}_{k-1} \\ \hat{\boldsymbol{m}}_{k-1} \end{bmatrix}, \qquad \bar{\boldsymbol{\Sigma}}_{k-1} = \begin{bmatrix} \boldsymbol{\Sigma}_{k-1}^{\boldsymbol{xx}} & \boldsymbol{\Sigma}_{k-1}^{\boldsymbol{xm}} \\ \boldsymbol{\Sigma}_{k-1}^{\boldsymbol{mx}} & \boldsymbol{\Sigma}_{k-1}^{\boldsymbol{mm}} \end{bmatrix}. \tag{7.27}$$

The estimate, $\hat{\boldsymbol{x}}_{k-1}$, of the vehicle state vector, $\boldsymbol{x}_{k-1}$, the corresponding covariance matrix, $\boldsymbol{\Sigma}_{k-1}^{\boldsymbol{xx}}$, and the control, $\boldsymbol{u}_{k-1}$, can then be utilized to evaluate the following standard UKF

Algorithm 5 Online simultaneous localization and mapping

```
1: initialize vehicle state vector estimate and covariance matrix
2: repeat
3:     UKF pred.: update vehicle state and covariance with vehicle dynamics model
4:     UKF pred.: update environment state and covariance with environment model
5:     UKF corr.: update vehicle state and covariance with vehicle output measurement
6:     for $i = 0, ..., m-1$ do
7:         if $i^{\text{th}}$ environment measurement corresponds to existing feature then
8:             set $j$ equal to index of existing feature that $i^{\text{th}}$ measurement corresponds to
9:         else
10:            extend state vector and covariance matrix
11:            initialize new state and covariance components using $i^{\text{th}}$ measurement
12:            set $j$ equal to index of new feature that $i^{\text{th}}$ measurement corresponds to
13:        UKF corr.: update state and covariance with $i^{\text{th}}$ environment measurement
14:    let system evolve for one sampling instant
```

update equations [87, 51]

$$\hat{\boldsymbol{X}}_{k-1} = \hat{\boldsymbol{x}}_{k-1}\mathbf{1}^{1\times(2n+1)} + \gamma^{\boldsymbol{x}} \left[\mathbf{0}^{n\times 1} \quad \boldsymbol{L}^{\boldsymbol{xx}}_{k-1} \quad -\boldsymbol{L}^{\boldsymbol{xx}}_{k-1}\right], \tag{7.28}$$

$$\hat{\boldsymbol{X}}'_k = \boldsymbol{f}(\hat{\boldsymbol{X}}_{k-1}, \boldsymbol{u}_{k-1}), \tag{7.29}$$

$$\hat{\boldsymbol{x}}'_k = \hat{\boldsymbol{X}}'_k \boldsymbol{w}^{\boldsymbol{x}}_m, \tag{7.30}$$

$$\boldsymbol{\Sigma}'^{\boldsymbol{xx}}_k = \left(\hat{\boldsymbol{X}}'_k - \hat{\boldsymbol{x}}'_k\mathbf{1}^{1\times(2n+1)}\right) \boldsymbol{W}^{\boldsymbol{x}}_c \left(\hat{\boldsymbol{X}}'_k - \hat{\boldsymbol{x}}'_k\mathbf{1}^{1\times(2n+1)}\right)^\top + \boldsymbol{\Theta}_k. \tag{7.31}$$

The above equations contain the symbols $\mathbf{1}$ and $\mathbf{0}$ representing matrices of size as specified, where all elements equal one or zero, respectively. The matrix $\boldsymbol{L}^{\boldsymbol{xx}}_{k-1}$ is found from a Cholesky decomposition of the corresponding covariance matrix, $\boldsymbol{\Sigma}^{\boldsymbol{xx}}_{k-1} = \boldsymbol{L}^{\boldsymbol{xx}}_{k-1}\boldsymbol{L}^{\boldsymbol{xx}}_{k-1}{}^\top$, and $\gamma^{\boldsymbol{x}} = \alpha\sqrt{n+\kappa}$ with the tuning parameters α and κ. Conceptually, equation (7.28) generates a matrix, $\hat{\boldsymbol{X}}_{k-1}$, containing $2n+1$ so-called sigma points in its columns that approximately describe the probability distribution of $\boldsymbol{x}_{k-1}$ as a Gaussian distribution. The vehicle model from (7.11) is used in (7.29) to propagate these sigma points, where a common shorthand notation is used to indicate the evaluation of $\boldsymbol{f}$ column by column for each individual sigma point. The propagated sigma points contained in the matrix $\hat{\boldsymbol{X}}'_k$ are then used in equations (7.30) and (7.31) to update the mean and covariance estimates of the vehicle state. The column vector of weights $\boldsymbol{w}^{\boldsymbol{x}}_m$ and the weighting matrix $\boldsymbol{W}^{\boldsymbol{x}}_c = \text{diag}(\boldsymbol{w}^{\boldsymbol{x}}_c)$ are chosen in the usual way involving the additional tuning parameter β, see [95].

In line 4 of algorithm 5 an analogous update to the environment state estimate, $\hat{\boldsymbol{m}}_{k-1}$, is applied for the general case of a time-varying environment map,

$$\hat{\boldsymbol{M}}_{k-1} = \hat{\boldsymbol{m}}_{k-1}\boldsymbol{1}^{1\times(4n_m+1)} + \gamma^{\boldsymbol{m}}\left[\boldsymbol{0}^{2n_m\times 1} \quad \boldsymbol{L}^{\boldsymbol{mm}}_{k-1} \quad -\boldsymbol{L}^{\boldsymbol{mm}}_{k-1}\right], \tag{7.32}$$

$$\hat{\boldsymbol{M}}'_k = \boldsymbol{p}(\hat{\boldsymbol{M}}_{k-1}), \tag{7.33}$$

$$\hat{\boldsymbol{m}}'_k = \hat{\boldsymbol{M}}'_k\boldsymbol{w}^{\boldsymbol{m}}_m, \tag{7.34}$$

$$\boldsymbol{\Sigma}'^{\boldsymbol{mm}}_k = \left(\hat{\boldsymbol{M}}'_k - \hat{\boldsymbol{m}}'_k\boldsymbol{1}^{1\times(4n_m+1)}\right)\boldsymbol{W}^{\boldsymbol{m}}_c\left(\hat{\boldsymbol{M}}'_k - \hat{\boldsymbol{m}}'_k\boldsymbol{1}^{1\times(4n_m+1)}\right)^\top + \Xi_k. \tag{7.35}$$

In the above equations, set $\gamma^{\boldsymbol{m}} = \alpha\sqrt{2n_m+\kappa}$ and $\boldsymbol{w}^{\boldsymbol{m}}_m$ and $\boldsymbol{W}^{\boldsymbol{m}}_c$ are again chosen in the standard way [95]. As above, $\boldsymbol{\Sigma}^{\boldsymbol{mm}}_{k-1} = \boldsymbol{L}^{\boldsymbol{mm}}_{k-1}\boldsymbol{L}^{\boldsymbol{mm}\top}_{k-1}$ can be found from a Cholesky decomposition.

The next step in line 5 of algorithm 5 is concerned with applying a UKF correction update to the vehicle state by using the measurement information $\boldsymbol{y}_k$. Again, Cholesky decomposition is deployed to obtain $\boldsymbol{\Sigma}'^{\boldsymbol{xx}}_k = \boldsymbol{L}'^{\boldsymbol{xx}}_k\boldsymbol{L}'^{\boldsymbol{xx}\top}_k$and the standard UKF correction update reads

$$\hat{\boldsymbol{X}}''_k = \hat{\boldsymbol{x}}'_k\boldsymbol{1}^{1\times(2n+1)} + \gamma^{\boldsymbol{x}}\left[\boldsymbol{0}^{n\times 1} \quad \boldsymbol{L}'^{\boldsymbol{xx}}_k \quad -\boldsymbol{L}'^{\boldsymbol{xx}}_k\right], \tag{7.36}$$

$$\hat{\boldsymbol{Y}}_k = \boldsymbol{g}(\hat{\boldsymbol{X}}''_k), \tag{7.37}$$

$$\hat{\boldsymbol{y}}_k = \hat{\boldsymbol{Y}}_k\boldsymbol{w}^{\boldsymbol{x}}_m, \tag{7.38}$$

$$\boldsymbol{\Sigma}^{\boldsymbol{yy}}_k = \left(\hat{\boldsymbol{Y}}_k - \hat{\boldsymbol{y}}_k\boldsymbol{1}^{1\times(2n+1)}\right)\boldsymbol{W}^{\boldsymbol{x}}_c\left(\hat{\boldsymbol{Y}}_k - \hat{\boldsymbol{y}}_k\boldsymbol{1}^{1\times(2n+1)}\right)^\top + \boldsymbol{\Phi}_k, \tag{7.39}$$

$$\boldsymbol{\Sigma}^{\boldsymbol{xy}}_k = \left(\hat{\boldsymbol{X}}''_k - \hat{\boldsymbol{x}}'_k\boldsymbol{1}^{1\times(2n+1)}\right)\boldsymbol{W}^{\boldsymbol{x}}_c\left(\hat{\boldsymbol{Y}}_k - \hat{\boldsymbol{y}}_k\boldsymbol{1}^{1\times(2n+1)}\right)^\top, \tag{7.40}$$

$$\hat{\boldsymbol{x}}''_k = \hat{\boldsymbol{x}}'_k + \boldsymbol{\Sigma}^{\boldsymbol{xy}}_k\left(\boldsymbol{\Sigma}^{\boldsymbol{yy}}_k\right)^{-1}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k), \tag{7.41}$$

$$\boldsymbol{\Sigma}''^{\boldsymbol{xx}}_k = \boldsymbol{\Sigma}'^{\boldsymbol{xx}}_k - \boldsymbol{\Sigma}^{\boldsymbol{xy}}_k\left(\boldsymbol{\Sigma}^{\boldsymbol{yy}}_k\right)^{-1}\left(\boldsymbol{\Sigma}^{\boldsymbol{xy}}_k\right)^\top. \tag{7.42}$$

Above, sigma points are generated in (7.36) and propagated through the measurement function in (7.37). Subsequently, in (7.38) and (7.39) the expected output value and its covariance are computed using the propagated sigma points. Additionally, in (7.40) the generated sigma points are utilized to approximate the cross-covariance between vehicle state and measurement output. Finally, equations (7.41) and (7.42) update the vehicle state estimate and its covariance using the actual measurement, $\boldsymbol{y}_k$, as well as the computed covariance matrices.

So far, the vehicle state was updated based on its dynamics model and its output measurement. On the other hand, the map was updated with its prediction model if applicable. Hence, the overall mean and covariance at this point can be composed as follows.

$$\hat{\bar{\boldsymbol{x}}}^0_k = \begin{bmatrix}\hat{\boldsymbol{x}}''_k \\ \hat{\boldsymbol{m}}'_k\end{bmatrix}, \qquad \bar{\boldsymbol{\Sigma}}^0_k = \begin{bmatrix}\boldsymbol{\Sigma}''^{\boldsymbol{xx}}_k & \boldsymbol{\Sigma}^{\boldsymbol{xm}}_{k-1} \\ \boldsymbol{\Sigma}^{\boldsymbol{mx}}_{k-1} & \boldsymbol{\Sigma}'^{\boldsymbol{mm}}_k\end{bmatrix}. \tag{7.43}$$

Note that the operations so far have no influence on the cross-covariances, therefore these off diagonal blocks still have the time index $k-1$.

As a next step, the environment sensor measurements are indexed by i and incorporated into the estimation one measurement at a time. In order to resolve the data association issue [95], the following approach as summarized in lines 7 to 12 of algorithm 5 is chosen. If the measurement, $\boldsymbol{z}_k^i$, detects an object, a hypothetical landmark is introduced. The hypothetical landmark location is computed by inverting equation (7.16) and solving for the landmark location given the measurement and by approximating the vehicle state by its current estimate. Subsequently, the distance between the new hypothetical landmark and all existing landmarks is computed and the measurement is associated with the closest landmark. However, if the closest landmark is further away than a certain threshold, a new landmark is introduced and the state vector and covariance matrix are extended accordingly.

After addressing the issue of data association, the measurement $\boldsymbol{z}_k^i$ corresponds to the j^{th} landmark. Next, a UKF measurement update based on this measurement information is performed as follows,

$$\hat{\bar{\boldsymbol{X}}}_k^i = \hat{\bar{\boldsymbol{x}}}_k^i \boldsymbol{1}^{1\times(2\bar{n}+1)} + \bar{\gamma}\left[\boldsymbol{0}^{\bar{n}\times 1} \quad \bar{\boldsymbol{L}}_k^i \quad -\bar{\boldsymbol{L}}_k^i\right], \tag{7.44}$$

$$\hat{\boldsymbol{Z}}_k^i = \boldsymbol{h}^j(\hat{\bar{\boldsymbol{X}}}_k^i), \tag{7.45}$$

$$\hat{\boldsymbol{z}}_k^i = \hat{\boldsymbol{Z}}_k^i \bar{\boldsymbol{w}}_m, \tag{7.46}$$

$$\bar{\boldsymbol{\Sigma}}_k^{\boldsymbol{zz},i} = \left(\hat{\boldsymbol{Z}}_k^i - \hat{\boldsymbol{z}}_k^i \boldsymbol{1}^{1\times(2\bar{n}+1)}\right)\bar{\boldsymbol{W}}_c\left(\hat{\boldsymbol{Z}}_k^i - \hat{\boldsymbol{z}}_k^i \boldsymbol{1}^{1\times(2\bar{n}+1)}\right)^\top + \boldsymbol{\Omega}_k, \tag{7.47}$$

$$\bar{\boldsymbol{\Sigma}}_k^{\bar{\boldsymbol{x}}\boldsymbol{z},i} = \left(\hat{\bar{\boldsymbol{X}}}_k^i - \hat{\bar{\boldsymbol{x}}}_k^i \boldsymbol{1}^{1\times(2\bar{n}+1)}\right)\bar{\boldsymbol{W}}_c\left(\hat{\boldsymbol{Z}}_k^i - \hat{\boldsymbol{z}}_k^i \boldsymbol{1}^{1\times(2\bar{n}+1)}\right)^\top, \tag{7.48}$$

$$\hat{\bar{\boldsymbol{x}}}_k^{i+1} = \hat{\bar{\boldsymbol{x}}}_k^i + \bar{\boldsymbol{\Sigma}}_k^{\bar{\boldsymbol{x}}\boldsymbol{z},i}\left(\bar{\boldsymbol{\Sigma}}_k^{\boldsymbol{zz},i}\right)^{-1}(\boldsymbol{z}_k^i - \hat{\boldsymbol{z}}_k^i), \tag{7.49}$$

$$\bar{\boldsymbol{\Sigma}}_k^{i+1} = \bar{\boldsymbol{\Sigma}}_k^i - \bar{\boldsymbol{\Sigma}}_k^{\bar{\boldsymbol{x}}\boldsymbol{z},i}\left(\bar{\boldsymbol{\Sigma}}_k^{\boldsymbol{zz},i}\right)^{-1}\left(\bar{\boldsymbol{\Sigma}}_k^{\bar{\boldsymbol{x}}\boldsymbol{z},i}\right)^\top. \tag{7.50}$$

Analogous to the previous UKF updates, $\bar{\boldsymbol{\Sigma}}_k^{\boldsymbol{xx},i} = \bar{\boldsymbol{L}}_k^i \bar{\boldsymbol{L}}_k^{i\top}$ is used. The setting for $\bar{\gamma}$ is $\bar{\gamma} = \alpha\sqrt{\bar{n}+\kappa}$ and $\bar{\boldsymbol{w}}_m$, $\bar{\boldsymbol{W}}_c$ are chosen as usual [95]. As indicated by line 6 of algorithm 5, the above equations are repeatedly applied. After the last loop iteration, set $\hat{\bar{\boldsymbol{x}}}_k = \hat{\bar{\boldsymbol{x}}}_k^m$ and $\bar{\boldsymbol{\Sigma}}_k = \bar{\boldsymbol{\Sigma}}_k^m$.

In summary, the filtering algorithm consists of a standard UKF applied to the vehicle state that performs a dynamics update and a measurement update using measurements related to the vehicle state. Also, the environment state estimate is updated if an environment prediction model is available. Subsequently, measurements from environment sensors are considered. For each measurement it is checked whether it corresponds to an existing or a new, previously unseen, landmark. The state and the covariance matrix are extended if necessary and a UKF measurement update is performed to include the measurement information.

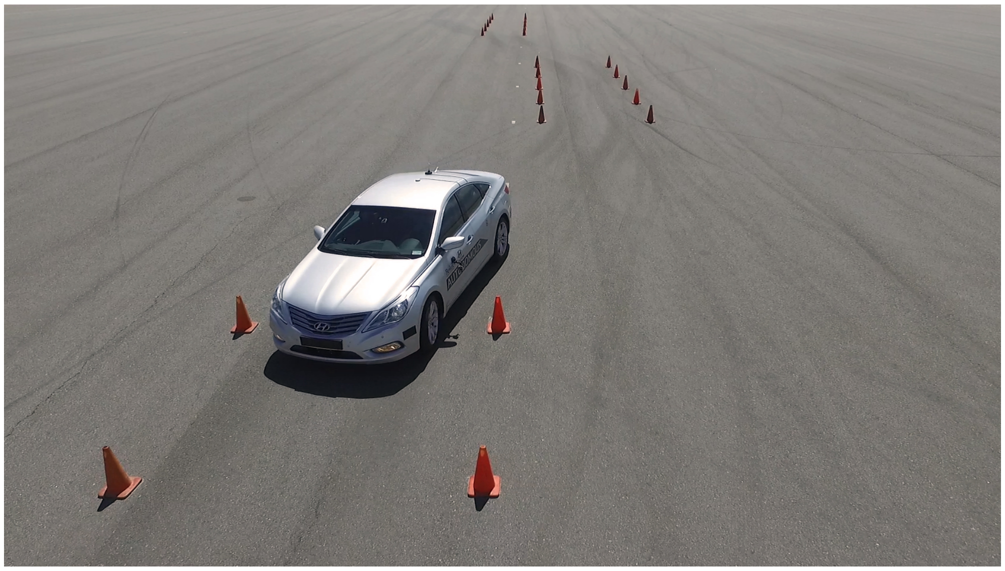

Figure 7.3: Experimental setup consisting of cones and the test vehicle entering the final segment of the test course

7.4 Experimental Setup

In the following, the application of the path following control and environment perception algorithms from section 7.3 for autonomously performing the ISO 3888-2 double lane change is discussed. Firstly, the considered test maneuver as well as the experimental vehicle are detailed. Subsequently, the overall control system architecture is explained.

7.4.1 Test Maneuver

The considered test scenario is the severe obstacle avoidance double lane change as given in the ISO 3888-2 norm [48]. The track consists of five segments, where the first, third and fifth are straight segments marked by five cones on each side of the track. The third segment has an offset with respect to sections one and five. The vehicle is required to change lanes within sections two and four. Furthermore, the vehicle is restricted to enter the course with a constant longitudinal velocity and the throttle is released 2 m after the beginning of the first cone segment. Figure 7.3 illustrates the experimental setup consisting of the cone course and the test vehicle, the latter will be described in more detail next.

7.4.2 Vehicle and Sensor Equipment

The test vehicle is a 5th generation Hyundai Grandeur, also marketed under the name Hyundai Azera. The vehicle is equipped with an aftermarket differential GPS with real time kinematic (RTK) precision enhancement, which yields position and heading informa-

GPS/IMU	
sensor type	Oxford RT3000
position accuracy	2×10^{-2} m
heading accuracy	1×10^{-1} deg
angular rate accuracy	0.2%
velocity accuracy	0.1%
LIDAR	
sensor type	IBEO LUX model 2010
range	200 m
accuracy	1×10^{-1} m
distance resolution	4×10^{-2} m
angular resolution	1.25×10^{-1} deg
max. scanning angle	45 deg
min. scanning angle	−45 deg

Table 7.1: Specifications for the considered inertial and GPS measurement system as well as the LIDAR sensors

tion with respect to a global reference frame. Velocities in the body fixed reference frame and angular rates are obtained from an inertial measurement unit. Details on the accuracy of the utilized Oxford RT3000 sensor unit are given in table 7.1.

The vehicle surroundings are perceived with six LIDAR sensors that are distributed on the perimeter of the vehicle. In particular, the sensor configuration is such that it gives a full 360 deg view of the environment. There is one front-facing and one rear-facing sensor, respectively, as well as side-facing sensors mounted in the front and rear on either side of the vehicle. Details of the utilized IBEO LUX model 2010 sensor are also included in table 7.1.

The experimental setup enables control authority over the steering angle and longitudinal acceleration. In particular, the power steering motor is utilized as an actuator for setting the desired steering angle. Note that a first order hold filter is applied to the desired steering signal before it is commanded to the actuator in order to avoid jerkily motion of the steering wheel. The desired acceleration is commanded through the factory adaptive cruise control interface. Online computations are performed on a DSPACE MicroAutoBox II real-time computer. This in-vehicle prototyping unit has a 900 MHz processor.

7.4.3 Control System Architecture

Figure 7.4 contains a schematic of the designed feedback control system. As shown in the figure, the control loop is closed around the plant, where the system boundary is chosen such that the plant contains the test vehicle, its sensor equipment, as well as the environment.

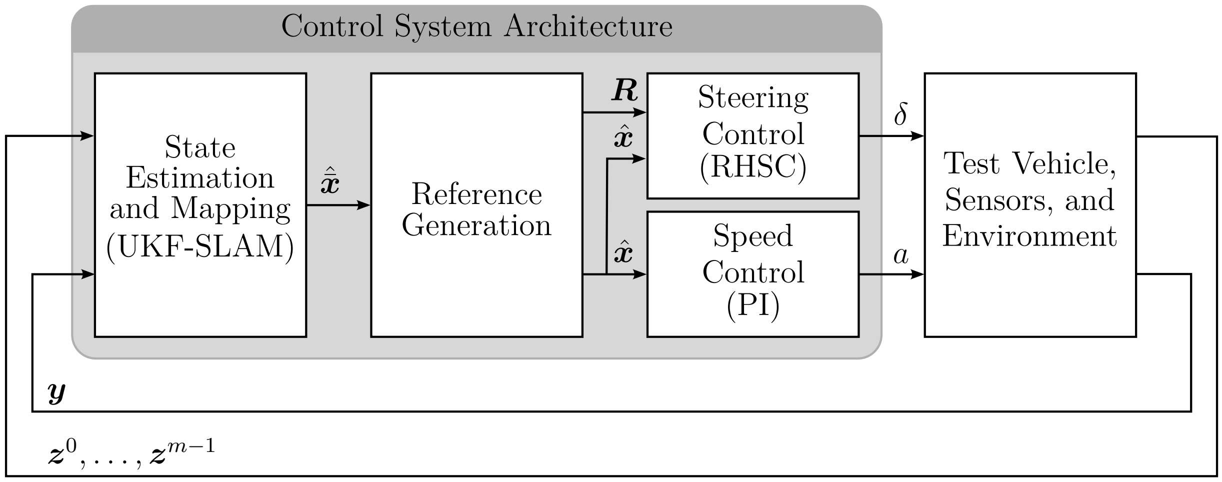
Control System Architecture
State Estimation and Mapping (UKF-SLAM)
$\hat{\bar{\boldsymbol{x}}}$
Reference Generation
$\boldsymbol{R}$
$\hat{\boldsymbol{x}}$
Steering Control (RHSC)
Speed Control (PI)
δ
a
Test Vehicle, Sensors, and Environment
$\boldsymbol{y}$
$\boldsymbol{z}^0, \dots, \boldsymbol{z}^{m-1}$

Figure 7.4: Block diagram of the autonomous vehicle feedback control loop with focus on the control system architecture

The overall control algorithm involves a perception, a path generation, as well as control modules. More details on each of these modules is given in the following.

Perception is performed in the UKF-SLAM block that contains the algorithm presented in section 7.3.2 using the models presented in section 7.2. In particular, since the test course is static in the considered case, set $\boldsymbol{m}_{k+1} = \boldsymbol{m}_k$ as the environment model. Since the environment is static, line 4 in algorithm 5, is unnecessary and therefore skipped for efficiency. Moreover, note that the stated UKF-SLAM equations (7.36) to (7.42) that correspond to line 5 of algorithm 5 hold for the general case when the measurement function $\boldsymbol{g}$ is nonlinear. However, the experimental setup allows direct measurement of all vehicle states and therefore the linear relation $\boldsymbol{g}(\boldsymbol{x}_k) = \boldsymbol{x}_k$ can be used and equations (7.36)-(7.42) are equivalently replaced with the standard Kalman filter measurement update to increase computational efficiency. The input to the UKF-SLAM algorithm are the measurements $\boldsymbol{y}$ and $\boldsymbol{z}^0, \dots, \boldsymbol{z}^{m-1}$ and the block computes an estimate of the overall state vector, $\hat{\bar{\boldsymbol{x}}}$, that contains the vehicle state estimate and a map of point landmarks representing cones.

Given a probabilistic map of the environment from the UKF-SLAM algorithm, a path is generated utilizing the following logic. Gates consisting of two cones are detected in the map and for each gate the middle point between the two cones is added to a list of waypoints. The waypoints are ordered in increasing X_O-direction as the vehicle is primarily moving in the X_O-direction. Then, a piecewise affine reference path through those waypoints is generated. Based on the planned control sequence from the previous time-step, future vehicle locations are computed and used together with the reference path for finding the reference sequence $\boldsymbol{R}$. This desired reference sequence and the estimated vehicle state vector are then passed to the path following RHSC and the speed controller.

The RHSC block computes the steering angle, δ, for following the desired reference sequence as specified by $\boldsymbol{R}$. The block involves solving the constrained finite-time optimal control problem (7.25) from section 7.3.1, where the models from section 7.2 are used for the vehicle dynamics and the nonlinear program is initialized with the current state estimate, $\hat{\boldsymbol{x}}$.

Longitudinal speed is controlled with a proportional-integral (PI) controller of the following form,

$$\begin{aligned} a_k &= \begin{cases} k_p(\dot{x}_k - \dot{x}_k^{\text{des}}) + k_i \sum_{\kappa=0}^{k-1}(\dot{x}_\kappa - \dot{x}_\kappa^{\text{des}}), \text{ if } X_k \leq 2\,\text{m} \\ 0, \text{ otherwise.} \end{cases} \\ &=: c(\boldsymbol{x}_k). \end{aligned} \tag{7.51}$$

The speed controller accelerates the vehicle to a desired target speed $\dot{x}^{\text{des}}$, subsequently, 2 m after entering the cone course the controller is turned off, which corresponds to releasing the throttle as required by the ISO standard [48].

7.5 Results

This section first presents parameter identification and tuning. Subsequently, the performance of the designed control system is evaluated with simulations and real-world experiments.

7.5.1 Parameter Identification and Tuning

The identified model parameters are given in table 7.2. The tire parameters were obtained using experimental data. Similarly, the air drag coefficient and the rolling resistance were determined experimentally using the coast down test [80]. Steering and steering rate constraints are chosen based on the physical limitations of the steering system and are also quantified in table 7.2.

In addition, table 7.2 contains the control system tuning. Note that the receding horizon sliding controller is tuned with the generic values $\eta = 1\,\text{m/rad}$ and $\rho = 0.9$. The prediction horizon is chosen to be $N = 15$ steps long, at a sampling time of $T_s = 100\,\text{ms}$ this results in a 1.5 s long preview window. The initial longitudinal speed upon entering the cone course is controlled to track $\dot{x}^{\text{des}} = 10\,\text{m/s}$. Finally, also the UKF-SLAM tuning is given in the table.

7.5.2 Simulations

Simulations were carried out in MATLAB/SIMULINK. More specifically, the controller is implemented in a C code S-function using the NPSOL software package [34] for nonlinear programming. The overall control logic consisting of SLAM, path generation, as well as speed and steering control is evaluated at a sampling time of $T_s = 100\,\text{ms}$. At the simulation stage the dynamics of the vehicle are emulated using the continuous-time equations (7.1), (7.2), (7.3), (7.4), (7.5), (7.6) and LIDAR measurements were emulated based on the geometry

Vehicle and Tire Parameters		
vehicle mass	M	1830.59 kg
vehicle rotational inertia	I_z	3477 kgm^2
distance from COG to front axle	l_f	1.1521 m
distance from COG to rear axle	l_r	1.6929 m
air density	d_a	1.225
air drag coefficient	C_d	0.2838
frontal area of vehicle	A_f	2.348
rolling resistance	R_x	186.8897
front tire coefficient	B_f	6.3846
front tire coefficient	C_f	1.5606
front tire coefficient	D_f	4.5385×10^3
front tire coefficient	E_f	-0.56
rear tire coefficient	B_r	13.2436
rear tire coefficient	C_r	1.5749
rear tire coefficient	D_r	3.2485×10^3
rear tire coefficient	E_r	0.4480
tire steering angle bound	$\bar{\delta}$	0.4333 rad
tire steering rate bound	$\bar{\dot{\delta}}$	0.0433 rad/s
Control System Parameters		
sampling time	T_s	100 ms
RHSC error dynamics parameter	ρ	0.9
RHSC error weight	η	1 m/rad
RHSC prediction horizon length	N	15
PI proportional gain	k_p	0.8
PI integral gain	k_i	0.02
UKF-SLAM parameter	α	0.5
UKF-SLAM parameter	β	2
UKF-SLAM parameter	κ	-1

Table 7.2: Identified vehicle parameters and chosen control system tuning

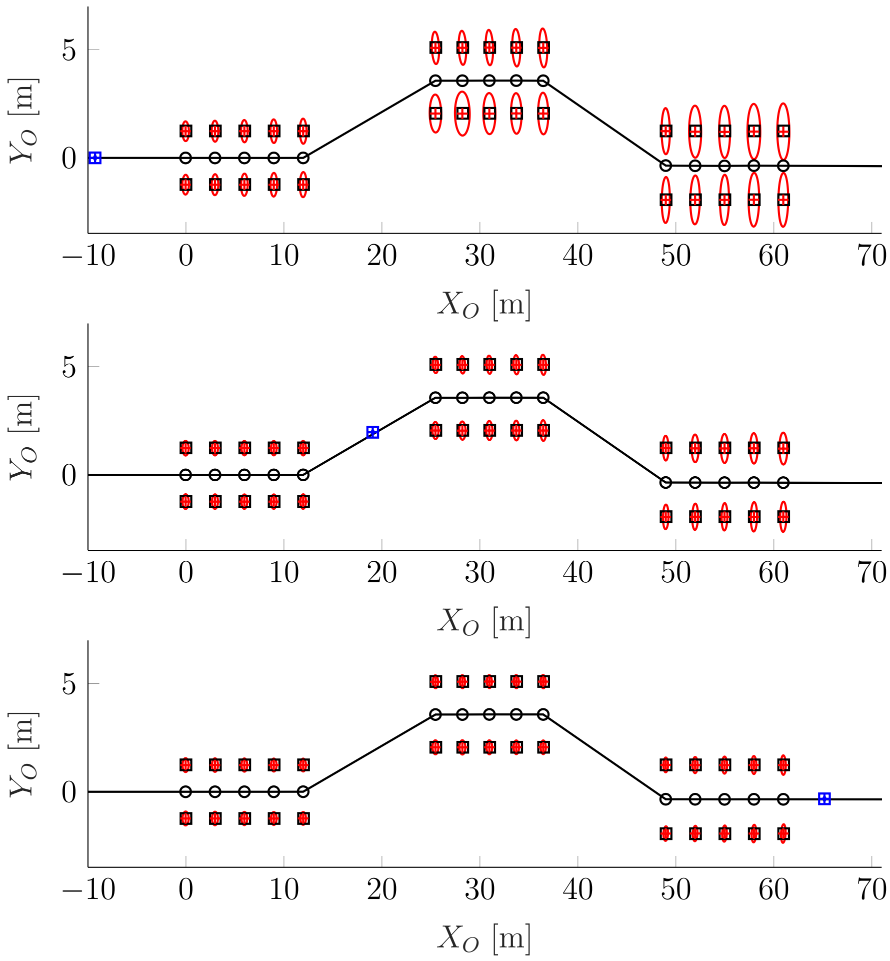

Figure 7.5: Exact cone locations (black, square), estimated cone locations (red, cross) and corresponding covariance ellipses (red, solid), generated waypoints (black, circle) and path (black, solid), vehicle location (blue, cross) and corresponding uncertainty ellipse (blue, solid) from a simulation scenario without process/measurement noise with the vehicle located at $X \approx -9\,\mathrm{m}$, $X \approx 19\,\mathrm{m}$, and $X \approx 65\,\mathrm{m}$

of the vehicle scenario. The noise terms $\boldsymbol{\theta}_k$, $\boldsymbol{\phi}_k$, and $\boldsymbol{\omega}_k$ were set to zero at the simulation stage. The Runge-Kutta 4 method with a fixed time-step of 10 ms is utilizes for numerically simulating the vehicle dynamics.

Figure 7.5 illustrates the performance of the UKF-SLAM filter. The figure shows the exact cone locations as well as their estimates and the covariance ellipses for snapshots taken at three different time-steps. The figure also includes the desired path, which was generated based on the estimated cone locations. The position of the vehicle's COG and the corresponding covariance ellipse is also included in the figure. It can be seen that the perception system performs well in simulations and objects are detected reliably.

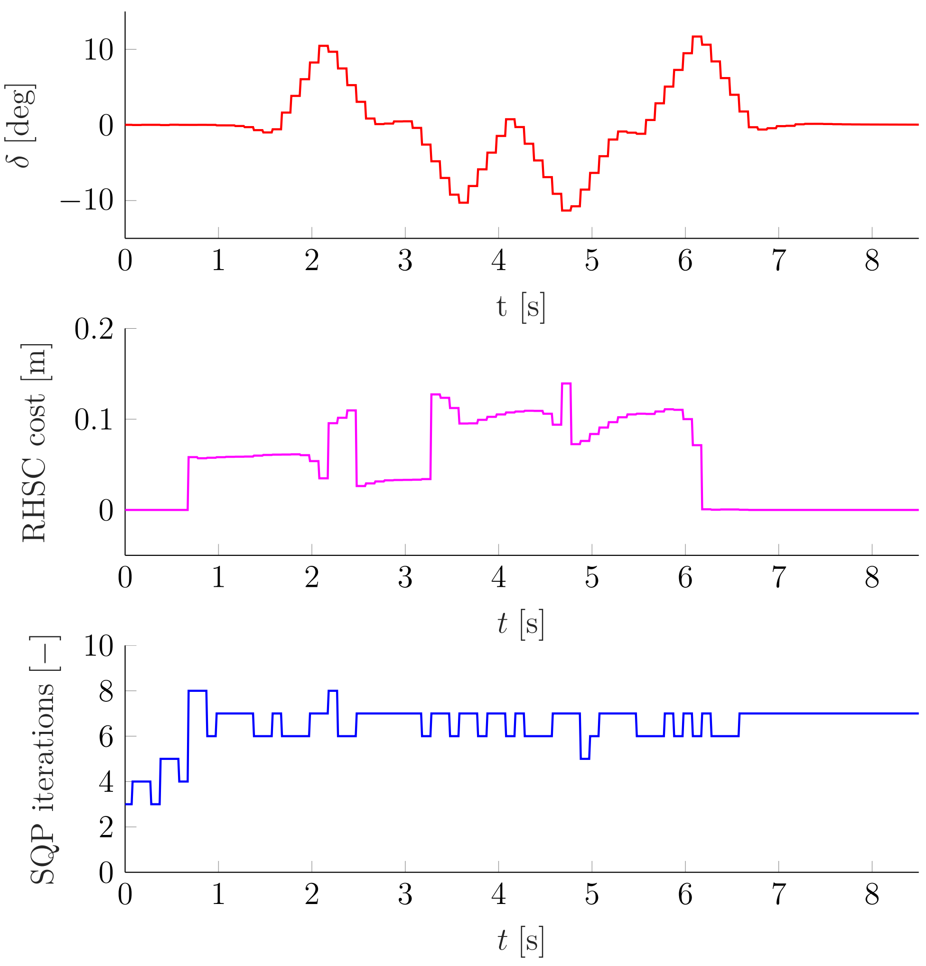

Figure 7.6: Commanded steering signal, residual cost after optimization, and utilized number of SQP iterations over simulation time from a scenario without process/measurement noise

While figure 7.5 shows performance of the UKF-SLAM algorithm, figure 7.6 shows characteristics of the RHSC logic. The computed steering signal along with the residual cost after minimization and the utilized number of sequential quadratic programming iterations is shown. It can be seen that the severity of the double lane change maneuver forces the RHSC to change the control signal rapidly and the steering rate constraint is frequently active. The figure also shows that the controller succeeds at keeping the cost close to zero throughout the simulation run. The relatively low number of less than ten SQP iterations per time-step causes the algorithm to execute very fast.

Figure 7.7 shows the closed-loop tracking performance of the designed control system. The true desired trajectory determined by the actual cone locations and the true vehicle location are plotted. Figure 7.8 gives more insight into the position and yaw tracking errors. It can be seen that the controller succeeds at keeping both errors close to zero. Note that

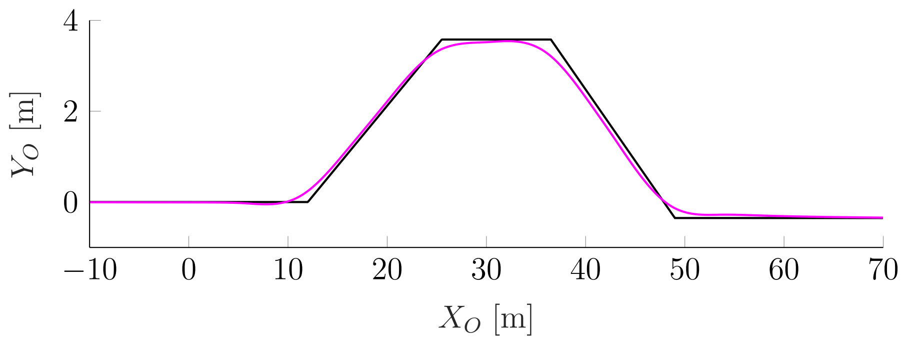

Figure 7.7: Desired path generated from the true cone locations (black, solid) and path traveled by the vehicle (magenta, solid) from a simulation scenario without process/measurement noise

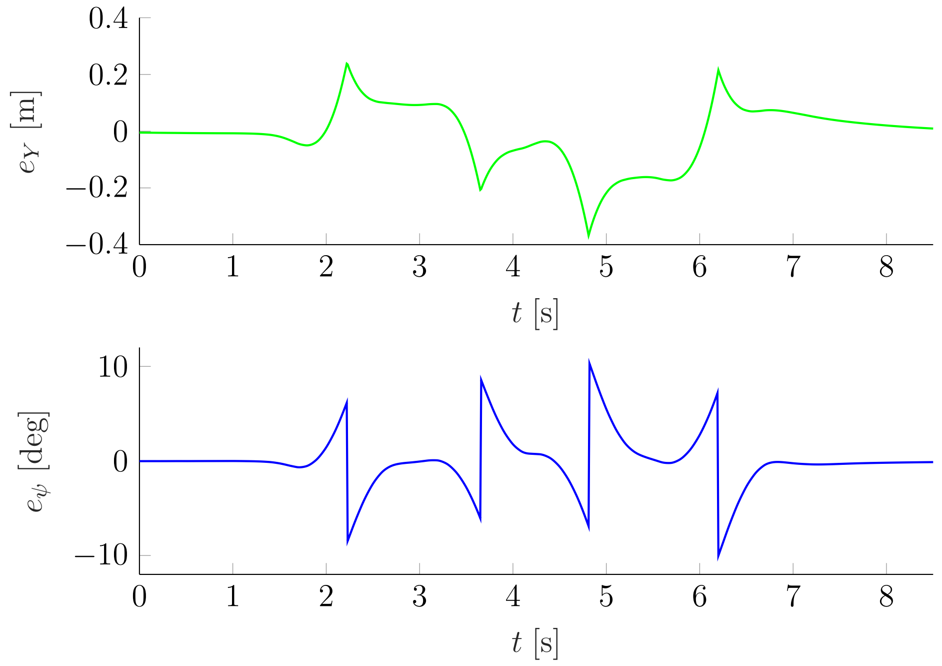

Figure 7.8: Position error and yaw tracking error over time from simulation without process/measurement noise

the non-smoothness/discontinuity of the plotted signals is a result of the non-smooth/discontinuous position and yaw references.

7.5.3 Experiments

Real-world experiments were carried out at the Hyundai-Kia Motors California Proving Grounds in California City, CA, USA. Automatic code generation is utilized to obtain the

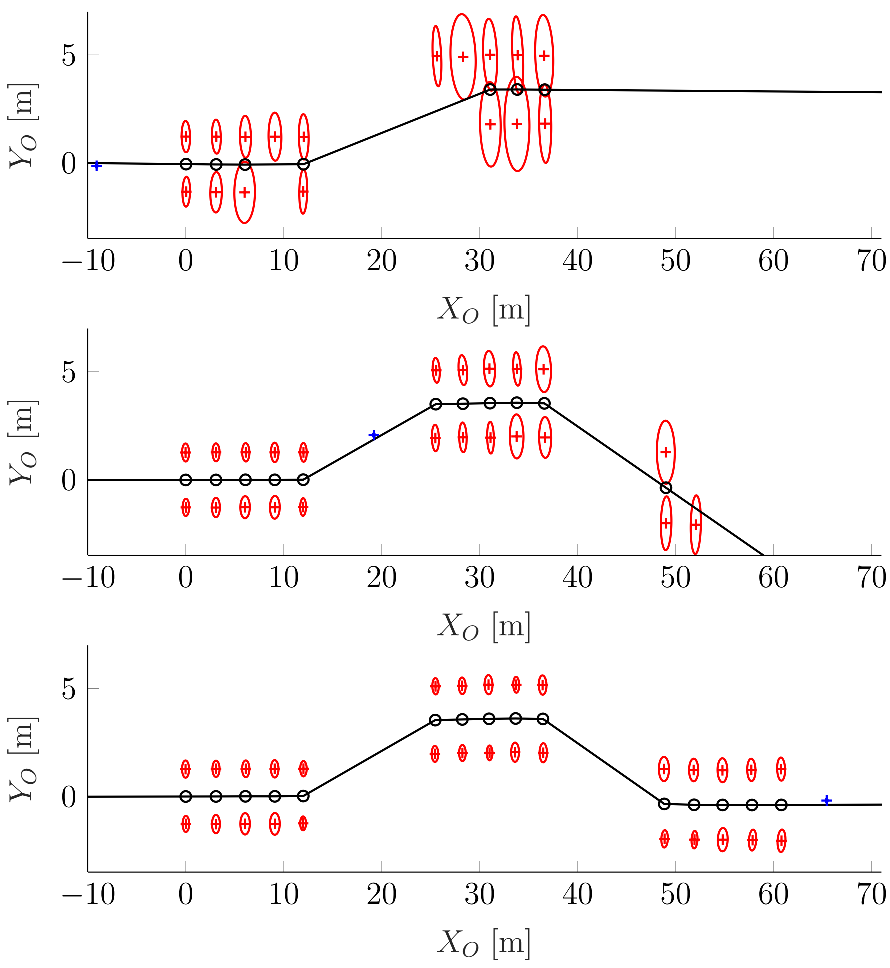

Figure 7.9: Estimated cone locations (red, cross) and corresponding covariance ellipses (red, solid), generated waypoints (black, circle) and path (black, solid), vehicle location (blue, cross) and corresponding uncertainty ellipse (blue, solid) from real-world experiments when the vehicle is roughly at $X \approx -9\,\mathrm{m}$, $X \approx 19\,\mathrm{m}$, and $X \approx 64\,\mathrm{m}$

controller software for the utilized DSPACE prototyping unit from the MATLAB/SIMULINK and C files described in the previous section. The tuning parameters are identical to those used in the simulation trials above.

Figure 7.9 illustrates the performance of the UKF-SLAM filter during experimental testing. Comparing the figure to the simulation results in figure 7.5 unveils that the distance range for reliable object detection with the LIDAR sensor is smaller than assumed in simulations. However, the algorithm still performs well enough to build a map and generate a path in time such that the overall control task is completed successfully.

Figure 7.10 shows the controller characteristics and it can be compared to figure 7.6 in the simulation case. The cost is slightly higher indicating slightly worse tracking performance

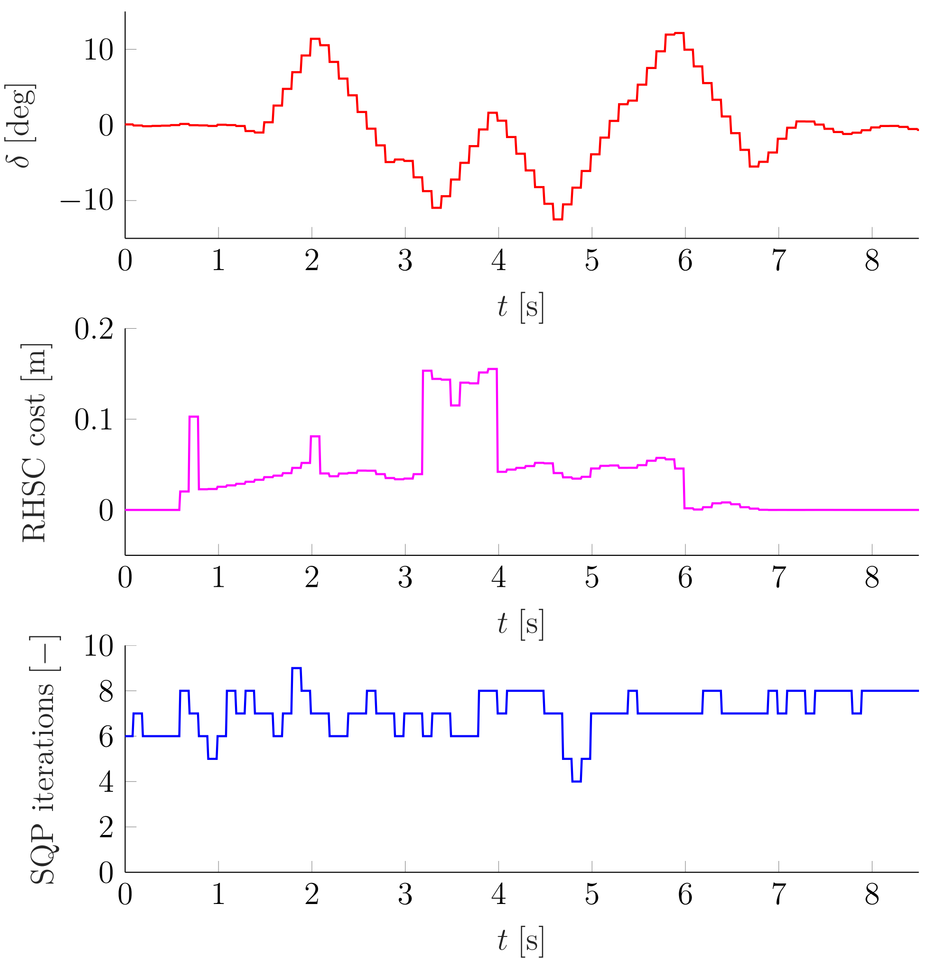

Figure 7.10: Commanded steering signal, residual cost after optimization, and utilized number of SQP iterations over time from real-world experiments

compared to the simulation case. This is expected due to the imperfection of the vehicle and sensor models yielding degraded performance in experiments. Similar to the simulation scenario iterations are rather low leading to fast evaluation of the controller software in real-time.

Figure 7.11 shows the path traveled by the vehicle in X_O, Y_O coordinates. It can be seen that there is a slight overshoot in section 5 of the cone course which was not present in the simulation case. However, the magnitude of the overshoot is small enough such that the maneuver can still be completed without collision with any of the cones.

In summary, the experimental results confirm the performance of RHSC in real-world experiments. Note that reference generation based on noisy feature locations yields a dynamically infeasible path. However, through its predictive control framework RHSC successfully compensates for this limitation making it an alternative to classical model predictive

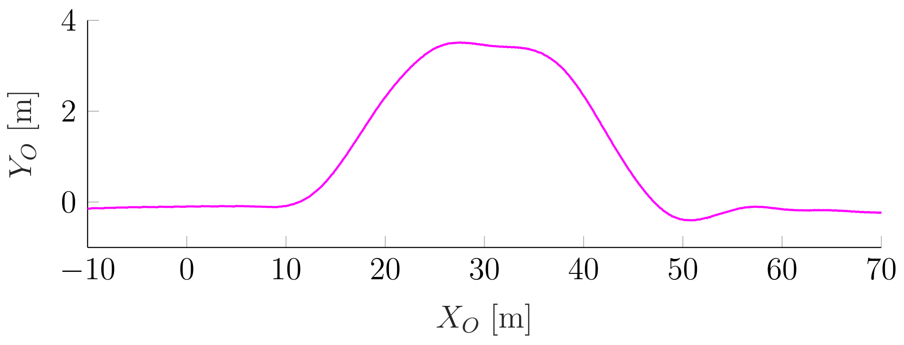

Figure 7.11: Path traveled by the vehicle (magenta, solid) from real-world experiments

control. The work shows that the use of desired error dynamics in the cost function, which is an uncommon choice in classical MPC, leads to good tracking performance. The eigenvalues of the error dynamics can be chosen to tune the aggressiveness of the controller while keeping the steering input as agile as possible by not penalizing it during the presented emergency maneuvers.

7.6 Concluding Remarks

This chapter deploys the receding horizon sliding control method for autonomously performing the severe obstacle avoidance lane change according to the ISO 3888-2 standard. Successful experiments show that the novel RHSC control method performs well in practical experiments and it is confirmed that the nonlinear programming solver can be executed in real-time despite the fast sampling time necessary for highly dynamic vehicle maneuvers. It is remarkable, that the controller tuning is extremely simple, there are only two tuning parameters involved and generic values lead to good performance both in simulations and real-world experiments. Results from this work imply that RHSC can be an alternative design approach to classical model predictive control for tracking references generated by high level motion planners in autonomous vehicles. Farther, the results confirm that it is possible to replace human test drivers by a control system in order to obtain objective and reproducible results in standardized tests. Hence, this work is relevant for ensuring that both autonomous vehicles as well as traditional human-driven vehicles meet sufficient stability levels even during severe maneuvers.

Part IV

Conclusion

Chapter 8

Closing Remarks

To close this dissertation, a summary of the presented work is provided in this chapter. Additionally, related results as well as potential future research paths are pointed out.

8.1 Summary of Results

This dissertation develops the receding horizon sliding control method, by merging model predictive control and sliding control. In particular, the resulting controller's objective is to minimize the deviation of the system state to sliding hypersurfaces using the receding horizon control framework. Sliding surfaces have desirable invariance properties, that are exploited for obtaining theoretical advancements related to state-of-the-art predictive control. Moreover, it is shown that the novel objective function yields good performance in practical case studies drawn from vehicle dynamics.

In particular, this dissertation first summarizes state-of-the-art model predictive control methodologies including predictive control for regulation problems for both linear and nonlinear systems. The relevance of invariant sets for guaranteeing persistent feasibility and stability is emphasized. Also, popular variants such as the delta input formulation are introduced. Moreover, it is explained how augmenting the state space by additional reference dimensions can be utilized for deriving provably stable and persistently feasible predictive controllers that allow for changing setpoints online.

Subsequently, this dissertation summarizes existing literature on combining model predictive control and sliding control. Then, the receding horizon sliding control method is developed for a wide class of constrained nonlinear systems. It is shown that the resulting cost function is a suitable Lyapunov-like function. Moreover, the well-known invariance property of sliding hyperplanes is revisited in a constrained setting, where a subset of the sliding surface remains invariant. Based on these two key observations, persistent feasibility and asymptotic stability are proven for the newly developed nonlinear predictive control scheme.

Furthermore, receding horizon sliding control is developed for linear tracking control with changing setpoints. State-of-the-art model predictive tracking controllers require an augmented state that includes a reference dimension in order to provide persistent feasibility and asymptotic stability. However, by exploiting the flatness property of sliding hyperplanes, it is shown that receding horizon sliding control can reduce the complexity of predictive tracking control to virtually the same complexity of predictive control for regulation problems.

The first application considered in this dissertation is a micro-underwater robot. This system's dynamic equations are high dimensional, nonlinear, and uncertain in practice. Applying receding horizon sliding control to the considered underwater robot yields a path following autopilot that achieves accurate tracking in the presence of model uncertainty and measurement noise. It is shown that the aggressiveness of the controller can be adjusted in an intuitive way to appropriately account for the uncertainty level.

Next, using an automotive cold-start control example, receding horizon sliding control is compared to the more established delta input formulation of classical model predictive control. A joint meta-cost function that takes into account tracking performance and computational characteristics is utilized to automatically tune both controllers. It is shown that receding horizon sliding control is of comparable computational complexity compared to classical predictive control and does not require additional computational resources. Hence, it is a viable alternative to classical predictive control that can potentially outperform its classical counterpart depending on the application at hand.

The practicality of receding horizon sliding control is further demonstrated by implementing it for steering control of an autonomous car. Specifically, receding horizon sliding control is integrated in a control-loop consisting of environment perception, path generation, and acceleration control for performing severe autonomous vehicle maneuvers. The computational effort remains manageable and allows application to the fast autonomous vehicle dynamics that require sampling times of the order of fractions of seconds. It is shown that the tuning effort of the designed receding horizon sliding controller is minimal and that it behaves well in simulations and real-world experiments of the ISO 3888-2 double lane change scenario.

8.2 Related Results

In addition to the results presented in this work, there have been additional studies that utilize receding horizon sliding control concepts [15, 71, 99]. A work that is particularly closely related to this dissertation is [63]. In this study, receding horizon sliding control is applied to an electric powertrain transmission. The system is described using a piecewise affine model that accounts not only for the contact mode but also for the backlash phase. A receding horizon sliding controller is designed for this hybrid system and the paper compares its performance to state-of-the-art predictive control strategies from the literature. The results indicate that the performance of the novel receding horizon sliding control approach is superior to classical predictive control for this particular electric vehicle transmission application. In particular, target wheel speed tracking accuracy is improved and the use of desired error dynamics in the controller formulation helps with smoothly re-establishing contact between the transmission elements after traversing the backlash mode. The latter point results in a considerable reduction of longitudinal vehicle jerk leading to enhanced ride comfort. Moreover, while the general structure of the optimization problems involved in both predictive controller formulations is similar, an average computation time reduction of roughly 16% is observed.

8.3 Outlook

The scope of this dissertation is relatively broad and spans the basic idea of receding horizon sliding control, the investigation of certain theoretical characteristics, and its application to selected real-world example applications including full-scale experiments. However, naturally there are still several aspects that are beyond the scope of this work that will be outlined in the following. These aspects can serve as potential future research directions.

Firstly, from a theoretical point of view the stability proofs presented in chapter 3 apply to a wide class of nonlinear systems. However, in practice the applicability of these results still depends on whether certain invariant set computations are tractable for a given system. This is a standard issue in nonlinear model predictive control. Further advancements in the area of invariant set computations for general nonlinear systems would be a seminal contribution to the nonlinear model predictive control field in general and would increase the impact of chapter 3 in particular.

Another potential research direction is the development of explicit receding horizon sliding control. Specifically for linear systems similar to those presented in chapter 4, multi-parametric programming [4] can be readily applied to obtain explicit receding horizon sliding controllers. It is an open question how receding horizon sliding control behaves in this context in terms of complexity and scalability when compared to classical model predictive control.

Furthermore, it was seen in chapters 5 and 7 that receding horizon sliding control keeps some of the desirable robustness properties of discrete sliding control. This is intuitive given that the controller attempts to minimize the distance of the system state to the sliding surface at every time-step. However, a complete analysis of the controller's robustness is beyond the scope of this work. It would be beneficial to gain further insight into the inherent robustness properties of the nominal receding horizon sliding controller and develop advanced formulations that are provably stable and persistently feasible in the presence of uncertainty.

The most pressing issue from a control practitioner point of view is a guideline on when the receding horizon sliding control method is preferable over classical model predictive control. At this point it is recommended to apply both methodologies and decide on a case-by-case basis which one performs better. For instance, in the engine control study presented in chapter 6 and in the transmission control paper [63] that was summarized in section 8.2, receding horizon sliding control outperforms classical model predictive control.

However, making reliable recommendations on which control methodology is superior in a given scenario, requires a far larger volume of case studies.

Similarly, it is desirable to gain deeper insight into the influence that the reshaped RHSC cost function has on different solver types and solver settings. While the general class of optimization problems solved is the same in both the RHSC and the MPC case, one control scheme may outperform the other in certain scenarios. For instance, in the case studies presented in chapter 6 and in [63] it was observed that RHSC resulted in lower computational complexity on average. A detailed analysis for the root cause for the tentatively lower computational load of RHSC, if one exists, would be worthwhile.

Bibliography

[1] Frank Allgöwer, Rolf Findeisen, and Zoltan K. Nagy. “Nonlinear Model Predictive Control: From Theory to Application”. In: *Journal of the Chinese Institute of Chemical Engineers* 35.3 (2004), pp. 299–315.

[2] Andrzej Bartoszewicz. “Discrete-time quasi-sliding-mode control strategies”. In: *IEEE Transactions on Industrial Electronics* 45.4 (1998), pp. 633–637.

[3] Alberto Bemporad, Allessandro Casavola, and Edoardo Mosca. “Nonlinear Control of Constrained Linear Systems via Predictive Reference Management”. In: *IEEE Transactions on Automatic Control* 42.3 (1997), pp. 340–349.

[4] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N. Pistikopoulos. “The explicit linear quadratic regulator for constrained systems”. In: *Automatica* 38.1 (2002), pp. 3–20.

[5] Wallace M. Bessa, Max S. Dutra, and Edwin Kreuzer. “An adaptive fuzzy sliding mode controller for remotely operated underwater vehicles”. In: *Robotics and Autonomous Systems* 58.1 (2010), pp. 16–26.

[6] Franco Blanchini and Stefano Miani. *Set-theoretic methods in control.* Birkhäuser Basel, 2015.

[7] Even Børhaug, Alexey Pavlov, and Kristin Y. Pettersen. “Integral LOS control for path following of underactuated marine surface vessels in the presence of constant ocean currents”. In: *47th IEEE Conference on Decision and Control (CDC), Dec. 9-11, Cancun, Mexico.* IEEE, 2008, pp. 4984–4991.

[8] Even Børhaug, Alexey Pavlov, and Kristin Y. Pettersen. “Straight line path following for formations of underactuated underwater vehicles”. In: *46th IEEE Conference on Decision and Control (CDC), Dec. 12-14, New Orleans (LA), USA*. IEEE, 2007, pp. 2905–2912.

[9] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems.* Cambridge University Press, 2017.

[10] Stephan Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge University Press, 2009.

[11] Eduardo F. Camacho and Carlos Bordons. “Nonlinear Model Predictive Control: An Introductory Review”. In: *Assessment and Future Directions of Nonlinear Model Predictive Control*. Ed. by Rolf Findeisen, Frank Allgöwer, and Lorenz T. Biegeler. Berlin, Heidelberg: Springer, 2007, pp. 1–16.

[12] H. Chen and F. Allgöwer. “A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability”. In: *Automatica* 34.10 (1998), pp. 1205–1217.

[13] Hai-Ying Chen, Shadab Mulla, Erich Weigert, Kenneth Camm, Todd Ballinger, Julian Cox, and Phil Blakeman. “Cold Start Concept (CSC™): A Novel Catalyst for Cold Start Emission Control”. In: *SAE International Journal of Fuels and Lubricants* 6 (2013), pp. 372–381.

[14] Jung Eun M. Choi, Shih-Yuan Liu, and J. Karl Hedrick. “Human Driver Model and Sliding Mode Control – Road Tracking Capability of the Vehicle Model”. In: *European Control Conference (ECC), 2015, July 15-17, Linz, Austria*. IEEE, 2015, pp. 2132–2137.

[15] Yongkeun Choi and J. Karl Hedrick. “Receding Horizon Sliding Control Design with Robustness to Implementation Imprecisions”. In: *2016 IEEE Conference on Control Applications*. 2016, pp. 1550–1555.

[16] M. Letizia Corradini and Guiseppe Orlando. “A VSC Algorithm Based on Generalized Predictive Control”. In: *Automatica* 33.5 (1997), pp. 927–932.

[17] E. J. Davison. “Some Properties of Minimum Phase Systems and ”Squared-Down” Systems”. In: *IEEE Transactions on Automatic Control* 28.2 (1983), pp. 221–222.

[18] S. S. Dughman and J. A. Rossiter. “A survey of guaranteeing feasibility and stability in MPC during target changes”. In: *9th International Symposium on Advanced Control of Chemical Processes, June 7-10, Whistler (BC), Canada*. IFAC, 2015, pp. 814–819.

[19] Kyle Edelberg, Selina Pan, and J. Karl Hedrick. “A discrete-time sliding mode formulation for automotive cold-start emission control”. In: *IEEE Conference on Decision and Control (CDC)*. IEEE, 2013, pp. 6818–6823.

[20] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. “Predictive Active Steering Control for Autonomous Vehicle Systems”. In: *IEEE Transactions on Control Systems Technology* 15.3 (2007), pp. 566–580.

[21] Dave Ferguson, Michael Darms, Chris Urmson, and Sascha Kolski. “Detection, Prediction, and Avoidance of Dynamic Obstacles in Urban Environments”. In: *IEEE Intelligent Vehicles Symposium, June 4-6, Eindhoven, The Netherlands*. IEEE, 2008, pp. 1149–1154.

[22] Daniel C. Fernández and Geoffrey A. Hollinger. “Model Predictive Control for Underwater Robots in Ocean Waves”. In: *IEEE Robotics and Automation Letters* 2.1 (2017), pp. 88–95.

[23] A. Ferramosca, D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. “MPC for tracking with optimal closed-loop performance”. In: *Automatica* 45 (2009), pp. 1975–1978.

[24] A. Ferramosca, D. Limon, I. Alvarado, T. Alamo, F. Castaño, and E. F. Camacho. “Optimal MPC for tracking of constrained linear systems”. In: *International Journal of Systems Science* 42.8 (2011), pp. 1265–1276.

[25] Thor I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, 2011.

[26] Thor I. Fossen, Morten Breivik, and Roger Skjetne. “Line-of-sight path following of underactuated marine craft”. In: *IFAC Proceedings Volumes* 36.21 (2003), pp. 211–216.

[27] Joseph Funke, Matthew Brown, Stephen M. Erlien, and J. Christian Gerdes. “Collision Avoidance and Stabilization for Autonomous Vehicles in Emergency Scenarios”. In: *IEEE Transactions on Control Systems Technology* 25.4 (2017), pp. 1204–1215.

[28] Katsuhisa Furuta. “Sliding mode control of a discrete system”. In: *Systems & Control Letters* 14.2 (1990), pp. 145–152.

[29] Weibing Gao, Yufu Wang, and Abdollah Homaifa. “Discrete-time variable structure control systems”. In: *IEEE Transactions on Industrial Electronics* 42.2 (1995), pp. 117–122.

[30] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, and J. Karl Hedrick. “Spatial Predictive Control for Agile Semi-Autonomous Ground Vehicles”. In: *Advanced Vehicle Control Conference, September 9-12 ,Seoul, South Korea*. AVEC, 2012.

[31] Winston Garcia-Gabin, Darine Zambrano, and Eduardo F. Camacho. “Sliding mode predictive control of a solar air conditioning plant”. In: *Control Engineering Practice* 17.6 (2009), pp. 652–663.

[32] Jorge L. Garriga and Masoud Soroush. “Model Predictive Control Tuning Methods: A Review”. In: *Industrial & Engineering Chemistry Research* 49.8 (2010), pp. 3505–3515.

[33] Andreas René Geist, Axel Hackbarth, Edwin Kreuzer, Viktor Rausch, Michael Sankur, and Eugen Solowjow. “Towards a Hyperbolic Acoustic One-Way Localization System for Underwater Swarm Robotics”. In: *IEEE International Conference on Robotics and Automation (ICRA), May 16-21, Stockholm, Sweden*. 2016, pp. 4551–4556.

[34] Philip E. Gill, Walter Murray, Michael A. Saunders, and Margaret H. Wright. *User’s Guide for NPSOL 5.0: A Fortran Package for Nonlinear Programming*. Tech. rep. Department of Management Science and Engineering, Stanford University, 1998.

[35] Arron Griffiths, Aleksandr Dikarev, Pete R. Green, Barry Lennox, Xavier Poteau, and Simon Watson. “AVEXIS–Aqua Vehicle Explorer for In-Situ Sensing”. In: *IEEE Robotics and Automation Letters* 1.1 (2016), pp. 282–287.

[36] Gurobi Optimization, Inc. *Gurobi Optimizer Reference Manual.* 2016. URL: `http://www.gurobi.com`.

[37] Axel Hackbarth, Edwin Kreuzer, and Eugen Solowjow. "HippoCampus: A micro underwater vehicle for swarm applications". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, Sept. 28-Oct. 2, Hamburg, Germany.* IEEE, 2015, pp. 2258–2263.

[38] Andreas Hansen and J. Karl Hedrick. "Nonlinear control design within the high level modeling framework for an engine cold start scenario". In: *2015 American Control Conference.* 2015, pp. 19–24.

[39] Andreas Hansen and J. Karl Hedrick. "Receding Horizon Sliding Control for Linear and Nonlinear Systems". In: *2015 American Control Conference.* 2015, pp. 1629–1634.

[40] Andreas Hansen, Yutong Li, and J. Karl Hedrick. "Invariant Sliding Domains for Constrained Linear Receding Horizon Tracking Control". In: *IFAC Journal of Systems and Control* 2 (2017), pp. 12–17.

[41] Martin Herceg, Michal Kvasnica, Colin N. Jones, and Manfred Morari. "Multi-Parametric Toolbox 3.0". In: *European Control Conference, July 17-19, Zurich, Switzerland.* IEEE, 2013, pp. 502–510.

[42] Rami Y. Hindiyeh and J. Christian Gerdes. "Equilibrium Analysis of Drifting Vehicles for Control Design". In: *ASME 2009 Dynamic Systems and Control Conference, October 12-14, Hollywood (CA), USA.* ASME, 2009, pp. 181–188.

[43] Ben Mansour Houda and Nouri Ahmed Said. "Discrete Predictive Sliding Mode Control of Uncertain Systems". In: *2012 9th International Multi-Conference on Systems, Signals and Devices (SSD), March 20-23, Chemnitz, Germany.* IEEE, 2012.

[44] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V. Kolmanovsky. "The Development of Model Predictive Control in Automative Industry: A Survey". In: *International Conference on Control Applications (CCA).* IEEE, 2012, pp. 295–302.

[45] Byron Thomas Shaw II. "Modelling and Control of Automotive Coldstart Hydrocarbon Emissions". PhD thesis. University of California, Berkeley, 2002.

[46] M. Innocenti and M. Falorni. "State constrained sliding mode controllers". In: *1998 American Control Conference.* 1998, pp. 104–108.

[47] A. Isidori. *Nonlinear control systems.* 3rd ed. Springer Berlin New York, 1995.

[48] *ISO 3888-2:2002(E): Passenger cars – Test track for a severe lane-change maneuvre – Part 2: Obstacle avoidance.* International Organization for Standardization. Geneva, CH, 2002.

[49] Tae-Jeong Jang, Hyun-Sik Ahn, and Chong-Ho Choi. "Iterative learning control for discrete-time nonlinear systems". In: *International Journal of Systems Science* 25.7 (1994), pp. 1179–1189.

[50] Tor A. Johansen. "Toward Dependable Embedded Model Predictive Control". In: *IEEE Systems Journal* 11.2 (2017), pp. 1208–1219.

[51] Simon J. Julier and Jeffrey K. Uhlmann. "Unscented Filtering and Nonlinear Estimation". In: *Proceedings of the IEEE* 92.3 (2004), pp. 401–422.

[52] Simon Julier and Jeffrey K. Uhlmann. *A General Method for Approximating Nonlinear Transformations of Probability Distributions.* Tech. rep. Department of Engineering Science, University of Oxford, 1996.

[53] J. Kim, S.-H. Oh, D. Cho, and J. K. Hedrick. "Robust discrete-time variable structure control methods". In: *Journal of Dynamic Systems, Measurement, and Control* 122 (2000), pp. 766–775.

[54] V. L. Kocic and G. Ladas. *Global Behavior of Nonlinear Difference Equations of Higher Order with Applications*. Kluwer Academic Publishers, 1993.

[55] A. V. Kraev, A. I. Rogovskii, and V. V. Fomichev. "On a Generalization of Relative Degree". In: *Differential Equations* 50.8 (2014), pp. 1122–1127.

[56] F. Kraus, B. D. O. Anderson, and M. Mansour. "Robust Schur polynomial stability and Kharitonov's theorem". In: *International Journal of Control* 47.5 (1988), pp. 1213–1225.

[57] Francoise Lamnabhi-Lagarrigue, Anuradha Annaswamy, Sebastian Engell, Alf Isaksson, Pramod Khargonekar, Richard M. Murray, Henk Nijmeijer, Tariq Samad, Dawn Tilbury, and Paul Van den Hof. "Systems & Control for the future of humanity, research agenda: Current and future roles, impact and grand challenges". In: *Annual Reviews in Control* 43 (2017), pp. 1–64.

[58] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *ROBOMECH Journal* 1.1 (2014).

[59] George Leitmann. "On one approach to the control of uncertain systems". In: *Journal of Dynamic Systems, Measurement, and Control* 115.2B (1993), pp. 373–380.

[60] Anastasios M. Lekkas. "Guidance and path-planning systems for autonomous vehicles". PhD thesis. Norwegian University of Science and Technology, 2014.

[61] Anastasios M. Lekkas and Thor I. Fossen. "A quaternion-based LOS guidance scheme for path following of AUVs". In: *IFAC Proceedings Volumes* 46.33 (2013), pp. 245–250.

[62] Anastasios M. Lekkas and Thor I. Fossen. "A time-varying lookahead distance guidance law for path following". In: *IFAC Proceedings Volumes* 45.27 (2012), pp. 398–403.

[63] Yutong Li, Andreas Hansen, J. Karl Hedrick, and Junzhi Zhang. "A receding horizon sliding control approach for electric powertrains with backlash and flexible halfshafts". In: *Vehicle System Dynamics* 55.12 (2017), pp. 1823–1841.

[64] Yi-Wen Liao and J. Karl Hedrick. “A Discrete-Time Integral Sliding Model Predictive Control for Obstacle Avoidance of Ground Vehicles”. In: *ASME 2013 Dynamic Systems and Control Conference, October 28-30, Columbus (OH), USA*. ASME, 2015.

[65] Yi-Wen Liao and J. Karl Hedrick. “Robust model predictive control with discrete-time integral sliding surface”. In: *2015 American Control Conference*. 2015, pp. 1641–1646.

[66] D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. “MPC for tracking piecewise constant references for constrained linear systems”. In: *Automatica* 44 (2008), pp. 2382–2387.

[67] D. Limon, A. Ferramosca, I. Alvarado, T. Alamo, and E. F. Camacho. “MPC for Tracking of Constrained Nonlinear Systems”. In: *Nonlinear Model Predictive Control*. Ed. by Lalo Magni, Davide Martino Raimondo, and Frank Allgöwer. Berlin Heidelberg: Springer-Verlag, 2009, pp. 315–323.

[68] Johan Löfberg. “YALMIP : A Toolbox for Modeling and Optimization in MATLAB”. In: *2004 IEEE Symposium on Computer Aided Control Systems Design, September 4, Taipei, Taiwan*. IEEE, 2004, pp. 284–289.

[69] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgöwer. “Robust model predictive control for nonlinear discrete-time systems”. In: *International Journal of Robust and Nonlinear Control* 13 (2003), pp. 229–246.

[70] Ruben Martinez-Cantin and José A. Castellanos. “Unscented SLAM for large-scale outdoor environments”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), August 2-6, Edmonton, Canada*. IEEE, 2005, pp. 3427–3432.

[71] Ahmad Mozaffari, Nasser L. Azad, Andreas Hansen, and J. Karl Hedrick. “A Receding Horizon Sliding Controller for Automotive Engine Coldstart: Design and Hardware-in-the-Loop Testing With an Echo State Network High-Fidelity Model”. In: *Asian Journal of Control* 18.4 (2016), pp. 1219–1238.

[72] So-Ryeok Oh and Jing Sun. “Path following of underactuated marine surface vessels using line-of-sight based model predictive control”. In: *Ocean Engineering* 37.2-3 (2010), pp. 289–295.

[73] Hans B. Pacejka. *Tire and Vehicle Dynamics*. SAE International, 2006.

[74] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. “A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 1.1 (2016), pp. 33–55.

[75] Selina Pan, Kyle Edelberg, and J. Karl Hedrick. “Discrete Adaptive Sliding Control of Automotive Powertrains”. In: *American Control Conference*. 2014, pp. 202–207.

[76] Mercedes Pérez de la Parte, Cristina M. Cirre, and Eduardo F. Camacho. “Application of Predictive Sliding Mode Controllers to a Solar Plant”. In: *IEEE Transactions on Control Systems Technology* 16.4 (2008), pp. 819–825.

[77] Ewa Pawłuszewicz and Zbigniew Bartosiewicz. “Euler’s discretization and dynamic equivalence of nonlinear control systems”. In: *Nonlinear control in the year 2000 volume 2*. Ed. by Alberto Isidori, Francoise Lamnabhi-Lagarrigue, and Witold Respondek. London: Springer, 2001, pp. 183–191.

[78] M. Pérez, E. Jiménex, and E. F. Camacho. “Design of an explicit constrained predictive sliding mode controller”. In: *IET Control Theory & Applications* 4.4 (2010), pp. 552–562.

[79] Anthony Phillips, Diana Yanakiev, and Fangjun Jiang. “Managing Complexity in Large-Scale Control System Design”. In: *2004 American Control Conference*. 2004, pp. 4698–4703.

[80] Rajesh Rajamani. *Vehicle Dynamics and Control*. Springer US, 2006.

[81] H. Richter. “A multi-regulator sliding mode control strategy for output-constrained systems”. In: *Automatica* 47.10 (2011), pp. 2251–2259.

[82] H. Richter, B. D. O’Dell, and E. A. Misawa. “Robust positively invariant cylinders in constrained variable structure control”. In: *IEEE Transactions on Automatic Control* 52.11 (2007), pp. 2058–2069.

[83] Anthony Rossiter. “A global approach to feasibility in linear MPC”. In: *International Control Conference (ICC), August 30-September 1, Glasgow, Scotland, UK*. UKACC, 2006.

[84] Matteo Rubagotti, Davide Martino Raimondo, Antonella Ferrara, and Lalo Magni. “Robust model predictive control with integral sliding mode in continuous-time sampled-data nonlinear systems”. In: *IEEE Transactions on Automatic Control* 56.3 (2011), pp. 556–570.

[85] Mahdi Shahbakhti, Mohammad Reza Amini, Jimmy Li, Satoshi Asami, and J. Karl Hedrick. “Early Model-Based Design and Verification of Automotive Control System Software Implementations”. In: *Journal of Dynamic Systems, Measurement, and Control* 137.2 (2015).

[86] Khoshnam Shojaie and Alireza Mohammad Shahri. “Iterated Unscented SLAM Algorithm for Navigation of an Autonomous Mobile Robot”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), September 22-26, Nice, France*. IEEE, 2008, pp. 1582–1587.

[87] Jeffrey K. Uhlmann Simon J. Julier. “A new extension of the Kalman filter to nonlinear systems”. In: *SPIE 3068, Signal Processing, Sensor Fusion, and Target Recognition VI*. SPIE, 1997, pp. 182–193.

[88] D. Simon, J. Löfberg, and T. Glad. “Reference tracking MPC using dynamic terminal set transformation”. In: *IEEE Transactions on Automatic Control* 59.10 (2014), pp. 2790–2795.

[89] Hebertt Sira-Ramírez. “Non-linear discrete variable structure systems in quasi-sliding mode”. In: *International Journal of Control* 54.5 (1991), pp. 1171–1187.

[90] Hebertt Sira-Ramírez. “Sliding regimes in general nonlinear systems: a relative degree approach”. In: *International Journal of Control* 50.4 (1989), pp. 1487–1506.

[91] Jean-Jacques E. Slotine and Weiping Li. *Applied nonlinear control*. Prentice Hall International Inc., 1991.

[92] S. K. Spurgeon. “Hyperplane design techniques for discrete-time variable structure control systems”. In: *International Journal of Control* 55.2 (1992), pp. 445–456.

[93] Leo V. Steenson, Stephen R. Turnock, Alexander B. Phillips, Catherine Harris, Maaten E. Furlong, Eric Rogers, Liuping Wang, Kenneth Bodles, and Derek W. Evans. “Model predictive control of a hybrid autonomous underwater vehicle with experimental verification”. In: *Journal of Engineering for the Maritime Environment* 228.2 (2014), pp. 166–179.

[94] Shreyas Sudhakar, Andreas Hansen, and J. Karl Hedrick. “Algorithmic Performance of Receding Horizon Sliding Control for Engine Emission Reduction”. In: *2016 IEEE Conference on Control Applications*. 2016, pp. 1398–1403.

[95] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, Cambridge (MA), 2006.

[96] Vadim Utkin, Jürgen Guldner, and Jingxin Shi. *Sliding mode control in electromechanical systems*. CRC Press, 2009.

[97] A. Wächter and L. T. Biegler. “On the Implementation of a Primal-Dual Interior Point Filter Line Starch Algorithm for Large-Scale Nonlinear Programming”. In: *Mathematical Programming* 106.1 (2006), pp. 25–57.

[98] Yizhou Wang, Wenjie Chen, Masayoshi Tomizuka, and Badr N. Alsuwaidan. “Model Predictive Sliding Mode Control – For Constraint Satisfaction and Robustness”. In: *ASME 2013 Dynamic Systems and Control Conference, October 21-23, Palo Alto (CA), USA*. ASME, 2013.

[99] Lianhao Yin, Gabriel Ingesson, Rolf Johansson, Per Tunestål, and J. Karl Hedrick. “Nonlinear Air-Path Control of A Heavy-Duty Diesel Engine – A Receding Horizon Sliding Control Approach”. In: *2017 American Control Conference*. 2017, pp. 3619–3624.

[100] Xinghuo Yu and Okyay Kaynak. “Sliding-mode control with soft computing: a survey”. In: *IEEE Transactions on Industrial Electronics* 56.9 (2009), pp. 3275–3285.

[101] Jiansuo Zhou, Zhiyuan Liu, and Run Pei. “A New Nonlinear Model Predictive Control Scheme for Discrete-Time System Based on Sliding Mode Control”. In: *2001 American Control Conference*. 2001, pp. 3079–3084.

[102] Jiansuo Zhou, Zhiyuan Liu, and Run Pei. “Sliding Mode Model Predictive Control with Terminal Constraints”. In: *Proceedings of the 3rd World Congress on Intelligent Control and Automation, June 28 - July 2, Hefei, P.R. China*. IEEE, 2000.