Technische Universität München

Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik

Near Minimum-Time Predictive Reference Governors for Mechatronic Drive Systems

Alexander Markus Dötlinger

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr.-Ing. Hans-Georg Herzog

Prüfer der Dissertation:

1. Univ.-Prof. Dr.-Ing. Ralph Kennel
2. Univ.-Prof. Dr.-Ing. Mario Pacas, Universität Siegen

Die Dissertation wurde am 02.07.2015 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 26.01.2016 angenommen.

Für Elisabeth

Vorwort

Herrn Prof. Dr.-Ing. Ralph Kennel sei herzlich für die Betreuung der vorliegenden Arbeit und für die freundliche Aufnahme am Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik gedankt.

Weiterer Dank gilt Herrn Prof. Dr.-Ing. Mario Pacas von der Universität Siegen für das Interesse an meiner Arbeit und die Übernahme des Zweitgutachtens.

Herzlich bedanken möchte ich mich auch bei Jean-François Stumper, der mich während meines Studiums und auch im Laufe meiner Promotion sehr unterstützt hat. Gedankt sei auch Peter Landsmann und Petros Karamanakos für die vielen nützlichen Diskussionen und Anmerkungen rund um meine Arbeit. Aber auch außerhalb des universitären Umfelds habe ich Unterstützung erhalten – großer Dank gilt hier Gerald Schmid für das Interesse an meiner Forschung und Alois Unterholzner, der durch sein hilfreiches Feedback und mit seinem Fachwissen entscheidend zum Gelingen dieser Arbeit beigetragen hat.

Danken will ich auch all meinen Studenten. Besonders hervorheben will ich hierbei Florian Larcher, Lukas Leitner und Tino Müller, die wirklich hervorragende Leistungen erbracht haben.

Ich möchte mich beim gesamten Lehrstuhl-Team für die ausgesprochen angenehme Atmosphäre und die entstandenen Freundschaften bedanken – besonders hervorgehoben seien hier die gesamte „Mittwochs-Koch-Gruppe“ und meine Bürokollegen Sascha Kühl, Dirk Paulus und Peter Stolze.

Ein großes Dankeschön geht auch an meine Eltern, die mich immer unterstützt haben und mit dem Ermöglichen meines Studiums den Grundstein für meine Promotion gelegt haben.

Zuletzt will ich mich bei meiner Freundin Elisabeth für das entgegengebrachte Verständnis für mein „Projekt Promotion“ bedanken. Ihre stetige Unterstützung hat mir immer die notwendige Kraft gegeben – ihr sei diese Arbeit gewidmet.

Alexander Dötlinger

St. Johann in Tirol im Juli 2016

"*Prediction is very difficult, especially about the future.*"
Niels Bohr (1885–1962)

Kurzfassung

Die vorliegende Arbeit beschäftigt sich mit einem nichtlinearen Sollsignal-Vorfilter für mechatronische Antriebssysteme. Dieses Vorfilter stellt einen Zusatz zu einem existierenden Regler dar. Es verändert ein Sollsignal dahingehend, dass der unterlagerte Regler Systemgrenzen, wie beispielsweise Spannungs- und Stromgrenzen, einhält sowie voll ausnutzt, um ein schnelles Regelverhalten zu erreichen. Das Prinzip dieses Vorfilters orientiert sich an der modellprädiktiven Regelung, die es erlaubt, eine Kostenfunktion unter Einhaltung von Grenzen zu minimieren.

Abstract

This work deals with predictive reference governors for mechatronic drive systems. A predictive reference governor is an add-on device to an existing feedback controller and adapts a reference signal such that constraints of the underlying control system, for example voltage and current limitations, are respected and simultaneously fully exploited in order to achieve a fast transient response. The concept of a predictive reference governor is closely related to model predictive control, where a cost function is minimized in the presence of constraints.

Contents

1 Introduction

The steadily increasing demand for a higher productivity in many industrial processes puts the focus on advanced control strategies for mechatronic drive systems. These control strategies try to achieve minimum-time results, which means that a reference value (set-point) should be reached in the shortest time that is physically possible. However, constraints that limit the dynamic performance (e.g., actuator limitations) are the bottleneck in many control applications. Traditional linear control methods, for example the well-known proportional-integral-derivative (PID) controller, often lead to unsatisfactory results in presence of constraints. To achieve minimum-time or at least near minimum-time results, it is necessary to directly incorporate the respective constraints into the control algorithm in order to fully exploit *and* respect the constraints.

1.1 State of the Art

Every practical control system is subject to certain constraints—common constraints in electrical drive systems are voltage, current, speed and position constraints. The actuator, i.e. the inverter, operates with a limited voltage, whereas the current constraints result from thermal considerations. Speed and position constraints are dependent on the type of the electrical motor and on the application. A widely known control structure for the control of electrical drives—especially in industry—is cascaded control, which consists of an inner current/torque control loop and outer speed and position control loops (Ellis and Lorenz, 1999; Zirn, 2008). A cascaded control structure traditionally consists of PID controllers, or variants thereof, with limited outputs to account for constraints. A problem that arises if the output of a controller with an integral part is limited is so-called integrator wind-up, which can lead to large control errors (Gilbert and Kolmanovsky, 1995). Anti wind-up approaches aim to tackle this problem, but the resulting control performance is often not satisfactory for applications that require a precise and fast transient response (Maciejowski, 2002). Apart from the control performance, constraints can also affect the stability of a control system. Even a *stable* linear system that is subject to input constraints cannot be globally stabilized by a *linear* control law (Saberi et al., 1996). This means, in general, that

a nonlinear control law is necessary to achieve a stable control system and a good control performance (Bemporad et al., 2002b).

Common design compromises to prevent constraints from being exceeded are: Limitations of the reference, a reduced dynamic performance of the linear controller, and an increased saturation (Gilbert and Kolmanovsky, 1995). If a linear controller is tuned such that constraints are respected, for example, even for the worst-case reference step height, the dynamic performance is degraded for smaller step heights, as the controller is *linear*. Increasing the saturation through a more powerful actuator might be inefficient if the full input range is only needed for short amounts of time. A popular method—especially in the fields of computer numerical control (CNC) machining and in robotics—is to limit the first and even higher order derivatives of the reference signal in order to respect constraints (Kröger, 2010; Weck, 1995; Lambrechts et al., 2005). Nevertheless, the discussed methods all represent design compromises—constraints are respected but often not fully exploited. Therefore, the control results can be improved in terms of dynamic performance.

Control methodologies that try to overcome the design compromises are: Anti wind-up schemes, model predictive control (MPC), reference governors (RGs), and other nonlinear control methods. Even though anti wind-up schemes are widely used, they pose several drawbacks. Besides the often unsatisfactory control results, traditional anti-wind up schemes can only deal with input constraints. Anti-wind up schemes in the outer control loops in a cascaded control structure can only limit the *reference* values of the underlying controllers and therefore have only indirect influence on the state constraints. MPC is a promising approach that makes use of constrained optimization, which allows minimizing a previously defined cost function in presence of input and state constraints. Hence, MPC can inherently handle constraints. Furthermore, MPC can deal with multivariable systems (Rawlings and Mayne, 2009; Maciejowski, 2002). The flexibility of the MPC formulation allows to include various design goals; for example, robust model predictive control (RMPC) aims to improve the robustness against uncertain or varying plant parameters (Campo and Morari, 1987; Casavola et al., 2000a; Schuurmans and Rossiter, 2000). In general, the MPC optimization problem has to be solved in real-time at every controller sampling instant—this is the reason why MPC first gained importance in the process industry, where sampling times in the range of seconds or minutes are involved. Nevertheless, due to the steadily increasing computational power, in recent years, MPC also found its way to the field of electrical drives, where sampling times in the range of tens of microseconds are common (Linder et al., 2010; Bolognani et al., 2011). An RG is an add-on to a feedback controller (see Figure 1.1). The feedback controller stabilizes a possibly unstable system in absence of constraints, whereas the RG adapts a reference signal such that input and state constraints of the underlying

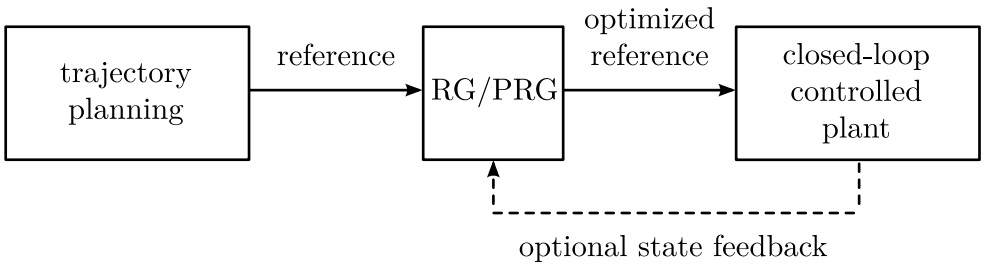


Figure 1.1: A basic RG/PRG structure

system are respected. If the reference signal can be tracked by the underlying controller without violating the constraints, the RG does not adapt the reference signal (Gilbert and Kolmanovsky, 1995; Kolmanovsky et al., 2012). In contrast to MPC, the RG structure allows to decouple the handling of constraints and the stabilization of a possibly unstable plant. A predictive reference governor (PRG) combines the advantages of the RG structure with the advantages of MPC, like the minimization of a cost function and the possibility to handle constraints (Bemporad and Mosca, 1995). An RG or a PRG can operate with or even without feedback (Sugie and Yamamoto, 2001; Hirata and Kogiso, 2001).

This section represents an overview of the state of the art of control methodologies for constrained systems. Further details and the corresponding references are given in the ongoing chapters of this thesis.

1.2 Problem Formulation and Contributions of this Thesis

The general goal of this work is to bring a control system to its physical limits. More precisely, it is intended to fully exploit input and state constraints to achieve a fast response. In order to directly incorporate constraints into a control scheme, the PRG approach, which relies on the ideas of MPC, is employed here. However, as the underlying optimization problem must usually be solved in real-time, this structure is not yet applicable to fast sampling systems. To overcome this drawback, a multi-rate PRG scheme is proposed, where the sampling time of the PRG can be higher than the sampling time of the feedback controller. In the extreme case, the PRG can be operated offline and without state feedback. This multi-rate scheme allows to exploit the features of MPC, like the handling of constraints and a flexible design of the cost function, even for fast sampling systems.

Typical MPC and PRG formulations use a quadratic form based cost function, which leads to a non-uniform weighting of high and low cost function values. Speaking in terms of a step response, this non-uniform weighting often leads to overshoots if a fast response is needed. These overshoots can be critical when precise and fast control results are needed. In this work, a PRG that relies on an ℓ_1-norm based cost function is proposed. The ℓ_1-norm (sum of

absolute values) ensures a uniform weighting and is able to decrease the overshoot in a step response. It is shown that an ℓ_1-norm based PRG leads to a response that is comparable with the minimum-time response, which represents the physical limit.

In order to improve the robustness of PRGs against uncertain or varying system parameters, it is proposed to use the ideas of RMPC for the design of robust PRGs. The robustness can be improved by the robust PRG formulation, even if no state feedback is available. Furthermore, the transient response is only marginally affected by using a robust PRG.

The ability to handle multivariable systems is demonstrated by a biaxial positioning example, where two electrical drives should follow a predefined path in two dimensions. It is shown that PRGs, like MPC, can inherently handle multivariable systems. Furthermore, the flexibility of the cost function is illustrated by PRGs that take into account the contour error, which is the main performance measure in biaxial positioning. An ℓ_1-norm based PRG that minimizes the contour error is proposed in order to achieve near minimum-time control results.

1.3 Outline

Chapter 2 introduces the concepts of convex optimization in order to efficiently formulate the optimization problems that arise in this thesis. Furthermore, the principles of MPC, RMPC, and RGs are discussed. The proposed multi-rate PRG control structure is presented in detail in Chapter 3. The system model and the feedback control structure of an industrial laser positioning system, which is used to validate the proposed control schemes, is introduced in Chapter 4. This laser positioning system consists of two permanent magnet (PM) direct current (DC) motors and is operated with a sampling time of 10 µs. The proposed near minimum-time PRG is validated by experimental results in Chapter 5. It is compared with a quadratic form based PRG, with an industry related standard method, and with the minimum-time solution that originates from a minimum-time optimal control problem. Chapter 6 introduces the robust PRG formulation that is based on RMPC and provides extensive experimental validation. The ability to handle multivariable systems is emphasized in Chapter 7, where PRGs are used for biaxial contouring. Furthermore, it is shown that the contour error, which is the main performance measure, can be incorporated in a PRG cost function. The proposed schemes are again validated by the industrial laser positioning system. Finally, conclusions and possible future research directions are given in Chapter 8.

2 Background

This chapter sets the foundation for an efficient formulation of the algorithms that are developed in this thesis; subsequent chapters often refer to the background information that is given here. As this work extensively makes use of optimization, the most important types of constrained convex optimization problems are discussed in Section 2.1. The control algorithms that are presented in this work rely on the ideas of model predictive control (MPC), so the most important principles and properties of MPC are presented in Section 2.2. The robust counterpart of MPC, namely robust model predictive control (RMPC), is described in Section 2.3 in order to introduce some concepts to increase the robustness against uncertain parameters. The presented MPC and RMPC schemes are in turn based on convex optimization problems. To complete this background section, the so-called reference governor (RG) is discussed in Section 2.4. An RG can be considered to be an add-on device that ensures constraint satisfaction, whereas the underlying feedback controller ensures—in absence of constraints—closed-loop stability of the plant.

2.1 Convex Optimization

This section provides the mathematical background for the control approaches that are presented in this thesis. Convex optimization problems are a subclass of mathematical optimization problems that play an important role in many fields, for example in finance, statistics, signal processing, and—most important for this work—control engineering. The goal of mathematical optimization is to determine an optimization variable such that a cost function is minimized while constraints, which are represented by equalities and/or inequalities, are respected. Due to the convexity of an optimization problem, any local minimum is also a *global* minimum; in contrast to non-convex optimization problems, where theoretically *all* local minima have to be evaluated to find the global minimum. This key property of convex optimization allows solving a problem in a reliable and efficient way.

Well-known convex optimization problems are, for example, least squares optimization problems, which can be solved analytically. Another example is a linear program (LP), which requires a numerical algorithm, a so-called solver, to solve the constrained optimization

problem. However, in recent decades other types of convex optimization problems, like quadratic programs (QPs) or second-order cone programs (SOCPs) gained attention (Boyd and Vandenberghe, 2004). As for LPs, numerical solvers are required to solve QPs and SOCPs.

Some important optimization problems are defined in this section in order to refer to them later on when the respective control algorithms are discussed. Furthermore, some useful transformations are given in order to formulate the problems that arise in this work in a standard form that is used by the respective LP, QP, or SOCP solver. The notation used in this section is detached from the rest of this work and is therefore not listed in the symbols section[1].

2.1.1 Convex Optimization Problem Definition

A function $f(\boldsymbol{x}) : \mathbb{R}^m \mapsto \mathbb{R}$ is convex if it satisfies

$$f(a\boldsymbol{x_1} + b\boldsymbol{x_2}) \leq af(\boldsymbol{x_1}) + bf(\boldsymbol{x_2}), \tag{2.1}$$

for all $\boldsymbol{x_1} \in \mathbb{R}^m$, $\boldsymbol{x_2} \in \mathbb{R}^m$ and all $a \in \mathbb{R}$, $b \in \mathbb{R}$. Furthermore, $a + b = 1$, $a \geq 0$, and $b \geq 0$ must hold. The function $f(x) = x^2$ serves as an example: For this parabola in the Euclidean space, the inequality (2.1) requires that two points on the graph, $(x_1, f(x_1))$ and $(x_2, f(x_2))$, can be connected by a line segment that lies above the graph in the interval between x_1 and x_2. This requirement is satisfied by the function $f(x) = x^2$, which implies its convexity.

Another property, which is often needed in convex optimization, is the affinity of a function. A function $f(\boldsymbol{x}) : \mathbb{R}^m \mapsto \mathbb{R}^n$ is affine if it has the form of

$$f(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}, \tag{2.2}$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{b} \in \mathbb{R}^n$.

A general convex optimization problem can be stated as

$$\begin{aligned} \underset{\boldsymbol{\xi}}{\text{minimize}} \quad & f_0(\boldsymbol{\xi}) \\ \text{subject to} \quad & f_i(\boldsymbol{\xi}) \leq 0, \quad i \in \{1, 2, \ldots, m_i\}, \\ & \boldsymbol{a}_j^\top \boldsymbol{\xi} = b_j, \quad j \in \{1, 2, \ldots, m_j\}, \end{aligned} \tag{2.3}$$

[1] The optimization problems that are discussed in this section set the mathematical foundation for the specific optimization problems that arise in this work. The notation for these specific problems is in turn listed in the symbols section.

where $\boldsymbol{\xi} \in \mathbb{R}^{m_\xi}$ is the optimization variable and $f_0(\boldsymbol{\xi}) : \mathbb{R}^{m_\xi} \mapsto \mathbb{R}$ is the cost function. The functions $f_i(\boldsymbol{\xi}) : \mathbb{R}^{m_\xi} \mapsto \mathbb{R}$ define the inequality constraints $f_i(\boldsymbol{\xi}) \leq 0$ and the conditions $\boldsymbol{a}_j^\top \boldsymbol{\xi} = b_j$ (where $\boldsymbol{a}_j \in \mathbb{R}^{m_\xi}$ and $b_j \in \mathbb{R}$ are known parameters) are called equality constraints.

Conditions for the convexity of the optimization problem (2.3) are (Boyd and Vandenberghe, 2004, p. 137):

- The cost function $f_0(\boldsymbol{\xi})$ is convex,
- the inequality constraint functions $f_i(\boldsymbol{\xi})$ are convex, and
- the equality constraints $\boldsymbol{a}_j^\top \boldsymbol{\xi} = b_j$ are affine in $\boldsymbol{\xi}$ (this is true for constant problem parameters $\boldsymbol{a}_j$ and $\boldsymbol{b}_j$).

The solution of the convex optimization problem (2.3) is the minimizer $\boldsymbol{\xi}^\star$, which minimizes the cost function $f_0(\boldsymbol{\xi})$ while the inequality and equality constraints are met. If it is not possible to find a solution, the problem is infeasible. Infeasibility might arise, for example, if conflicting constraints are chosen. As (2.3) is convex, the minimizer $\boldsymbol{\xi}^\star$ represents a *global* optimum if the problem is feasible (Boyd and Vandenberghe, 2004, p. 138). This fact is a key property of convex optimization problems. Contrary to this, non-convex optimization problems might have several *local* minima that have different cost function values. These several local minima can make solving the optimization problem time-consuming. It might be hard or even impossible to determine *all* local minima in order to find the *global* minimum, which is usually of interest. A problem that often occurs in non-convex optimization problems is that a solver gets stuck in a local minimum.

In the following, some standard convex optimization problems that are important in the field of control engineering are stated and briefly discussed.

2.1.2 Linear Programming

A linear program (LP) with the optimization variable $\boldsymbol{\xi} \in \mathbb{R}^{m_\xi}$ can be expressed as

$$\begin{aligned} \underset{\boldsymbol{\xi}}{\text{minimize}} \quad & \boldsymbol{b}_0^\top \boldsymbol{\xi} + a_0 \\ \text{subject to} \quad & \boldsymbol{F}_1 \boldsymbol{\xi} \leq \boldsymbol{b}_1, \\ & \boldsymbol{F}_2 \boldsymbol{\xi} = \boldsymbol{b}_2, \end{aligned} \tag{2.4}$$

where $a_0 \in \mathbb{R}$, $\boldsymbol{b}_0 \in \mathbb{R}^{m_\xi}$, $\boldsymbol{b}_1 \in \mathbb{R}^{m_{b_1}}$, $\boldsymbol{b}_2 \in \mathbb{R}^{m_{b_2}}$, $\boldsymbol{F}_1 \in \mathbb{R}^{m_{b_1} \times m_\xi}$, and $\boldsymbol{F}_2 \in \mathbb{R}^{m_{b_2} \times m_\xi}$ define the LP and are problem specific (Boyd and Vandenberghe, 2004, p. 146). The less than or equal symbol "$\leq$" is used here to state component-wise inequalities, which means that every element of the vector $\boldsymbol{F}_1 \boldsymbol{\xi}$ must be less than or equal to the respective element of

the vector $\boldsymbol{b}_1$. Checking the conditions for convexity stated in Section 2.1.1 shows that the LP (2.4) is convex.

Linear programming is used in this work to minimize a cost function consisting of the sum of absolute values of a vector (ℓ_1-norm). How to cast an ℓ_1-norm based optimization problem as an LP is shown in a later part of this thesis.

2.1.3 Quadratic Programming

A quadratic program (QP) with the optimization variable $\boldsymbol{\xi} \in \mathbb{R}^{m_\xi}$ can be stated as

$$\begin{aligned} \underset{\boldsymbol{\xi}}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{\xi}^\top \boldsymbol{F}_0 \boldsymbol{\xi} + \boldsymbol{b}_0^\top \boldsymbol{\xi} + a_0 \\ \text{subject to} \quad & \boldsymbol{F}_1 \boldsymbol{\xi} \leq \boldsymbol{b}_1, \\ & \boldsymbol{F}_2 \boldsymbol{\xi} = \boldsymbol{b}_2, \end{aligned} \tag{2.5}$$

where $a_0 \in \mathbb{R}$, $\boldsymbol{b}_0 \in \mathbb{R}^{m_\xi}$, $\boldsymbol{b}_1 \in \mathbb{R}^{m_{b_1}}$, $\boldsymbol{b}_2 \in \mathbb{R}^{m_{b_2}}$, $\boldsymbol{F}_0 \in \mathbb{R}^{m_\xi \times m_\xi}$, $\boldsymbol{F}_1 \in \mathbb{R}^{m_{b_1} \times m_\xi}$, and $\boldsymbol{F}_2 \in \mathbb{R}^{m_{b_2} \times m_\xi}$ are the problem data (Boyd and Vandenberghe, 2004, p. 152). The constraints have the same form as in the LP (2.4); the cost function, however, is now quadratic. In order to get a convex cost function $\frac{1}{2}\boldsymbol{\xi}^\top \boldsymbol{F}_0 \boldsymbol{\xi} + \boldsymbol{b}_0^\top \boldsymbol{\xi} + a_0$, it is necessary that $\boldsymbol{F}_0$ is symmetric ($\boldsymbol{F}_0 = \boldsymbol{F}_0^\top$) and that $\boldsymbol{F}_0$ is positive definite ($\boldsymbol{F}_0 \succ 0$) (Boyd and Vandenberghe, 2004, p. 458). The LP (2.4) is a sub-problem of the stated QP, as (2.5) reduces to an LP if $\boldsymbol{F}_0 = \boldsymbol{0}$.

Quadratic programming is often used in the field of model predictive control (MPC), as QPs can be solved efficiently. Hence, QPs play an important role in this work, as the control algorithms that are presented are based on the ideas of MPC.

2.1.4 Second-Order Cone Programming

Another convex optimization problem, a so-called second-order cone program (SOCP) can be written as

$$\begin{aligned} \underset{\boldsymbol{\xi}}{\text{minimize}} \quad & \boldsymbol{b}_0^\top \boldsymbol{\xi} \\ \text{subject to} \quad & \|\boldsymbol{F}_i \boldsymbol{\xi} + \boldsymbol{c}_i\|_2 \leq \boldsymbol{b}_i^\top \boldsymbol{\xi} + a_i, \quad i \in \{1, 2, \ldots, m_i\}, \\ & \boldsymbol{F}_m \boldsymbol{\xi} = \boldsymbol{b}_m, \end{aligned} \tag{2.6}$$

where $a_i \in \mathbb{R}$, $\boldsymbol{b}_0 \in \mathbb{R}^{m_\xi}$, $\boldsymbol{c}_i \in \mathbb{R}^{m_{c_i}}$, $\boldsymbol{b}_m \in \mathbb{R}^{m_{b_m}}$, $\boldsymbol{b}_i \in \mathbb{R}^{m_\xi}$, $\boldsymbol{F}_i \in \mathbb{R}^{m_{c_i} \times m_\xi}$, and $\boldsymbol{F}_m \in \mathbb{R}^{m_{b_m} \times m_\xi}$ are the respective problem data (Boyd and Vandenberghe, 2004, p. 156). The optimization variable is $\boldsymbol{\xi} \in \mathbb{R}^{m_\xi}$.

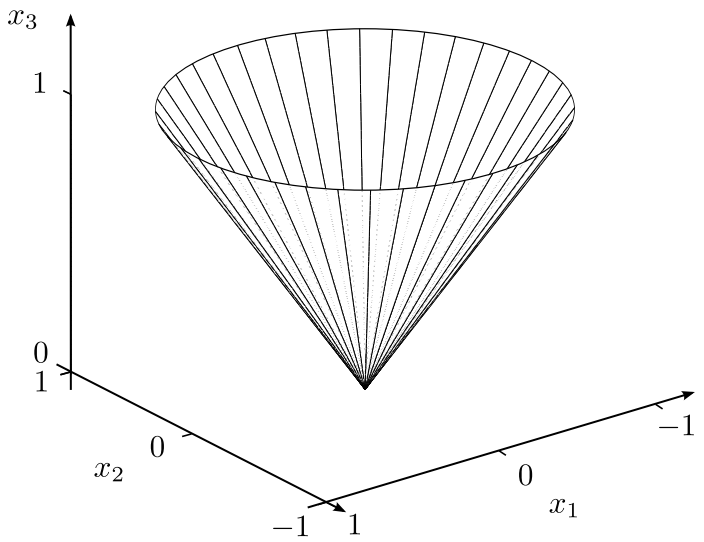


Figure 2.1: Graphical representation of the exemplary second-order cone constraint $\left\| \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix} \right] \right\|_2 \leq x_3$

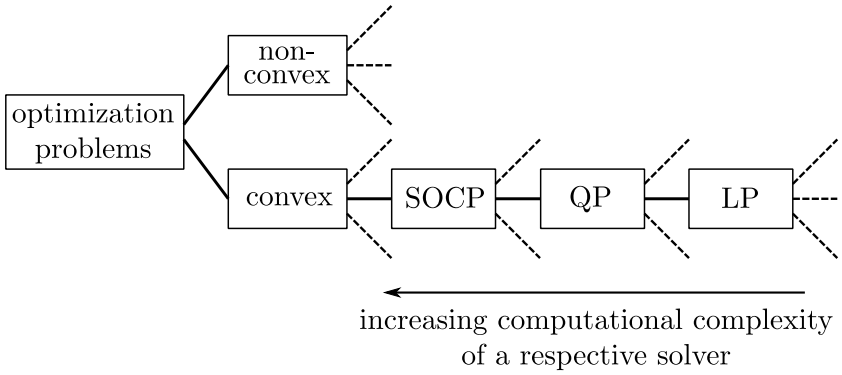


Figure 2.2: Relation between some convex optimization problems

This SOCP is again convex (see Boyd and Vandenberghe, 2004, p. 30, for details). An exemplary second-order cone is shown in Figure 2.1, which illustrates why second-order cones are also often called "ice-cream cones". SOCPs can arise, for example, in the field of robust model predictive control (RMPC), where the robustness of MPC against uncertain system parameters is improved.

LPs and QPs are sub-problems of SOCPs. A rough and not complete overview of convex optimization problems and their sub-problems is shown in Figure 2.2.

2.1.5 Useful Transformations

Some useful transformations for convex optimization problems are given in the following. The aim is to transform optimization problems that arise in the course of this work to convex standard forms (LPs, QPs, and SOCPs). It is desirable to express optimization problems in a standard form, as there exist reliable and efficient solvers for these types of problems.

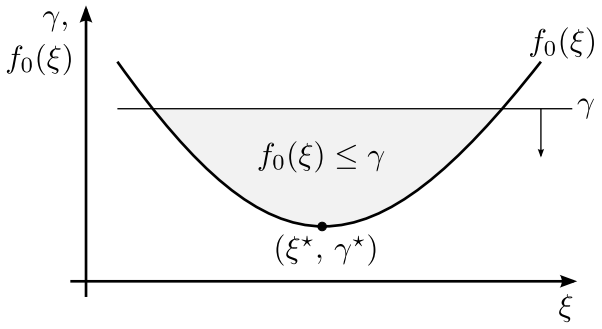


Figure 2.3: Graphical representation of the epigraph transformation for an optimization problem with a single optimization variable ξ and without constraints except the constraints introduced by the transformation. The feasible set (grey shaded) is expressed by $f_0(\xi) \leq \gamma$.

2.1.5.1 Epigraph Transformation

The epigraph transformation transforms the general convex optimization (2.3) such that the cost function becomes linear and the original cost function is expressed as a constraint. This transformation is used, for example, to formulate the RMPC optimization problem that arises in this work as an SOCP. Furthermore, the epigraph transformation can be used to show that a QP is a sub-problem of an SOCP.

Applying the epigraph transformation (Boyd and Vandenberghe, 2004, p. 134) to the general convex optimization problem (2.3) leads to

$$\begin{aligned} \underset{\boldsymbol{\xi},\gamma}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & f_0(\boldsymbol{\xi}) \leq \gamma, \\ & f_i(\boldsymbol{\xi}) \leq 0, \quad i \in \{1, 2, \ldots, m_i\}, \\ & \boldsymbol{a}_j^\top \boldsymbol{\xi} = \boldsymbol{b}_j, \quad j \in \{1, 2, \ldots, m_j\}, \end{aligned} \tag{2.7}$$

where an additional optimization variable $\gamma \in \mathbb{R}$ is introduced by the epigraph transformation. The original cost function $f_0(\boldsymbol{\xi})$ in (2.3) is now located in the constraints section of (2.7); the other constraints remain the same. The new cost function only consists of the scalar γ.

Figure 2.3 illustrates the idea of the epigraph transformation. If γ is decreased—while respecting $f_0(\xi) \leq \gamma$—as far as possible, the optimal point $(\xi^\star, \gamma^\star)$ is found. This means that the same optimal value of $\xi^\star$ is found if the epigraph transformation is not used. Hence, the optimization problems (2.3) and (2.7) are equivalent.

2.1.5.2 Transformation of an ℓ_1-Norm based Cost Function to a Linear Program

The transformation of an ℓ_1-norm based optimization problem to an LP in standard form is important in this work.

The ℓ_1-norm of a vector $\boldsymbol{z} = \begin{bmatrix} z_1 & z_2 & \cdots & z_{m_z} \end{bmatrix}^\top \in \mathbb{R}^{m_z}$ is defined as

$$\|\boldsymbol{z}\|_1 = \sum_{i=1}^{m_z} |z_i| \,. \tag{2.8}$$

An optimization problem that aims at minimizing an ℓ_1-norm based cost function is stated as

$$\begin{aligned} \underset{\boldsymbol{\xi}}{\text{minimize}} \quad & \|\boldsymbol{F}_0\boldsymbol{\xi} + \boldsymbol{b}_0\|_1 \\ \text{subject to} \quad & \boldsymbol{F}_1\boldsymbol{\xi} \leq \boldsymbol{b}_1, \\ & \boldsymbol{F}_2\boldsymbol{\xi} = \boldsymbol{b}_2, \end{aligned} \tag{2.9}$$

where $\boldsymbol{b}_0 \in \mathbb{R}^{m_{b_0}}$, $\boldsymbol{b}_1 \in \mathbb{R}^{m_{b_1}}$, $\boldsymbol{b}_2 \in \mathbb{R}^{m_{b_2}}$, $\boldsymbol{F}_0 \in \mathbb{R}^{m_{b_0} \times m_\xi}$, $\boldsymbol{F}_1 \in \mathbb{R}^{m_{b_1} \times m_\xi}$, and $\boldsymbol{F}_2 \in \mathbb{R}^{m_{b_2} \times m_\xi}$ are problem specific data. The optimization problem (2.9) is convex, because the ℓ_1-norm of the affine function $\boldsymbol{F}_0\boldsymbol{\xi} + \boldsymbol{b}_0$ is convex (Boyd and Vandenberghe, 2004, p. 24). This constrained ℓ_1-norm based minimization problem can be cast as the LP (Boyd and Vandenberghe, 2004, p. 294)

$$\begin{aligned} \underset{\boldsymbol{\xi},\boldsymbol{\beta}}{\text{minimize}} \quad & \mathbf{1}_{m_{b_0}}^\top \boldsymbol{\beta} \\ \text{subject to} \quad & -\boldsymbol{\beta} \leq \boldsymbol{F}_0\boldsymbol{\xi} + \boldsymbol{b}_0 \leq \boldsymbol{\beta}, \\ & \boldsymbol{F}_1\boldsymbol{\xi} \leq \boldsymbol{b}_1, \\ & \boldsymbol{F}_2\boldsymbol{\xi} = \boldsymbol{b}_2, \end{aligned} \tag{2.10}$$

where the vector $\boldsymbol{\beta} \in \mathbb{R}^{m_{b_0}}$ is introduced by this transformation to an LP and is now—besides $\boldsymbol{\xi}$—part of the optimization variables. This transformation is illustrated in Figure 2.4. The basic idea of the transformation to an LP is related to the epigraph transformation. The LP (2.10) can in turn be written as

$$\begin{aligned} \underset{\begin{bmatrix}\boldsymbol{\xi}^\top & \boldsymbol{\beta}^\top\end{bmatrix}^\top}{\text{minimize}} \quad & \begin{bmatrix} \mathbf{0}_{m_\xi} \\ \mathbf{1}_{m_{b_0}} \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\beta} \end{bmatrix} \\ \text{subject to} \quad & \begin{bmatrix} \boldsymbol{F}_0 & -\boldsymbol{I}_{m_{b_0} \times m_{b_0}} \\ -\boldsymbol{F}_0 & -\boldsymbol{I}_{m_{b_0} \times m_{b_0}} \\ \boldsymbol{F}_1 & \mathbf{0}_{m_{b_1} \times m_{b_0}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\beta} \end{bmatrix} \leq \begin{bmatrix} -\boldsymbol{b}_0 \\ \boldsymbol{b}_0 \\ \boldsymbol{b}_1 \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{F}_2 & \mathbf{0}_{m_{b_2} \times m_{b_0}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\beta} \end{bmatrix} = \boldsymbol{b}_2, \end{aligned} \tag{2.11}$$

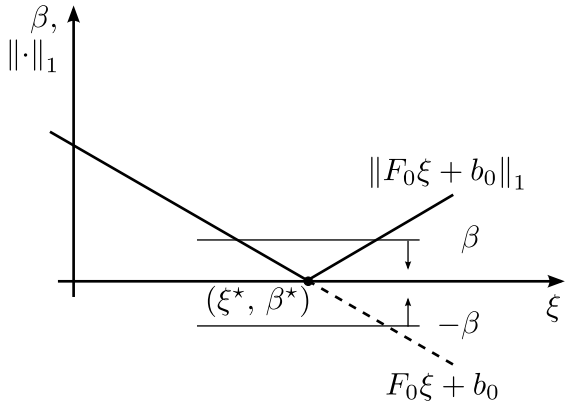


Figure 2.4: Illustration of the transformation of an ℓ_1-norm based cost function (see (2.9)) to a constraint in an LP (see (2.10)). This simple example shows the principle for scalar values of ξ, β, F_0 and b_0.

which is a standard form used by many LP solvers. The vector $\left[\, \boldsymbol{\xi}^\top \; \boldsymbol{\beta}^\top \,\right]^\top \in \mathbb{R}^{m_\xi + m_{b_0}}$ is the optimization variable.

Summing up, this section shows that the constrained optimization problem of minimizing the ℓ_1-norm of an affine function can be transformed to an LP.

2.1.6 Optimization Problem Solvers

Most convex optimization problems do not have an analytical solution—especially if constraints are considered. Therefore, numerical solvers have been developed to solve these problems.

LPs can be efficiently solved by using, for example, Dantzig's simplex method (Dantzig, 1965). Recent LP solvers are based on interior-point methods. In contrast to the simplex algorithm, interior-point algorithms try to find an optimum by passing through the interior of the feasible region. According to (Boyd and Vandenberghe, 2004), linear programming can be considered to be a "mature technology", which means that there are plenty of well-tested and efficient solvers available. The solver used here for LPs is called qpOASES (Ferreau et al., 2008) and is an active-set solver for QPs. It turned out that this QP solver, which can naturally handle LPs, is faster than Mathwork's MATLAB® LP solver linprog. The reason is qpOASES' efficient warm starting feature, which means that previous optimization result can be efficiently used as an initial value for the actual optimization process.

The QPs arising in this work are also solved with qpOASES (Ferreau et al., 2008) following the same reasoning as for LPs. qpOASES turned out to be even faster than the fast gradient algorithm of FiOrdOs (Ullmann, 2011) and Mathwork's MATLAB® QP solver quadprog.

When dealing with SOCPs here, the solver SeDuMi (Sturm, 1999) is used. Other SOCP solvers were presented, for example, by Domahidi (2013) or Mittelmann (2003).

Figure 2.2 on page 9 illustrates the computational complexity of the solvers for the respective optimization problems. A solver for SOCPs can be considered more complex than a solver for QPs, which is, in turn, more complex than a solver for LPs. This seems natural, as SOCPs include QPs and LPs as sub-problems.

Advanced methods in control engineering often involve optimization problems. A challenge is the need to solve the respective problem in *real-time*. This is especially the case if high sampling rates are needed due to the high dynamic requirements of the considered application. The controller sampling rates are normally in the kHz range for the control of electrical drives (Stumper et al., 2012) or even in the MHz range for the control of atomic force microscopes (Jerez et al., 2013). Hence, it becomes necessary to solve optimization problems within fractions of a second. Recently, fast gradient methods have been used to solve QPs on digital signal processors (DSPs) and field programmable gate arrays (FPGAs) with sampling rates of up to 1 MHz (Zometa et al., 2013; Jerez et al., 2013; Giselsson, 2013). Other methods that are capable of real-time optimization rely on automatic problem specific code generation (Ullmann, 2011; Mattingley and Boyd, 2012; Domahidi et al., 2013). Another approach for real-time optimization is multi-parametric programming (Herceg et al., 2013). The result of multi-parametric programming is an explicit control law (explicit MPC). The real-time task is reduced to the evaluation of a binary search-tree, where the optimal feedback control law is stored depending on the actual state in the state-space. There is even research in the field of analog optimization (Vichik and Borrelli, 2013, 2014) where the optimization problem is solved by an analog electronic circuit.

2.2 Model Predictive Control

The concept of model predictive control (MPC) has its roots in the 1960s (Propoi, 1963) and basically has been developed and used in industry for quite a long time before it gained importance in the academic field. MPC is a control methodology that determines the control inputs by solving a finite horizon open-loop optimal control problem at each sampling instant. Thereby, the actual state of the plant is used, as the initial state of this optimal control problem. MPC has several advantages compared to the widespread classic proportional-integral-derivative (PID) control and other standard control methods (Maciejowski, 2002; Camacho and Bordons, 2004):

- The underlying idea of MPC is easy to understand and offers great flexibility for a large number of control problems.
- System constraints can be directly incorporated in the formulation of the control problem. This is essential especially in the presence of safety constraints, or when operation

close to the constraints is desired, for example, in order to improve the dynamic performance.

- MPC can inherently handle multivariable control problems.

Of course, there are also drawbacks: Even though the idea of MPC is easy to understand, it might not be that simple to derive an optimization problem that leads to satisfying results. The design of MPC consists of the formulation of a cost function, the incorporation of constraints, and the tuning of respective weights in the cost function to achieve the desired control results. The real-time execution of MPC is another problem—the optimization problem has to be solved within one sampling interval. The real-time application can be a challenging task if low sampling times are involved. This is the reason why MPC first has been mainly used in the process industry, where slow system dynamics and the therefore low sampling times in the range of seconds or even minutes allow the real-time computation of the optimization problem (Maciejowski, 2002). However, in recent years, MPC has also gained importance in the field of power electronics and control of electrical drives, where sampling times in the range of tens of microseconds are common (Cortes et al., 2008; Linder et al., 2010; Stumper et al., 2012). A survey of industrial applications of MPC can be found in (Qin and Badgwell, 2003). Historical background about MPC is given in (Morari and H. Lee, 1999; Goodwin, 2012).

The following sections introduce the basic principles and concepts of MPC. As the control approaches that are presented in this work are strongly related to MPC, the foundation for the developed algorithms is set here.

2.2.1 Principle

The basic ingredients of MPC are a plant model, a cost function that should be minimized, and constraints that have to be respected. The plant model is needed to predict the system behavior over a certain time-span, the prediction horizon, in order to consider the *future* evolution of system states and system inputs in the cost function. In most of the cases, the predicted errors between the reference values and the system outputs are incorporated in the cost function. Bringing together the cost function and the system constraints leads to an optimization problem that has to be solved at every sampling instant. Another reason for the necessity to predict the system behavior results from the fact that—as *dynamic* systems are involved—it is normally not enough to just check the constraints for the actual sampling instant.

An important fact about MPC is that even applying the principle of MPC to a *linear* plant leads to a *nonlinear* control law if constraints are considered in the optimization problem (Mayne et al., 2000).

Figure 2.5 illustrates the basic principle of MPC using a simple example. Assuming that the actual time instant is k, the optimal input trajectory $\boldsymbol{U}^{\star}[k] = \left[\, \boldsymbol{u}^{\star}[k]^{\top} \; \ldots \; \boldsymbol{u}^{\star}[k+N-1]^{\top} \,\right]^{\top}$ is determined by solving a previously defined optimization problem in order to steer the system output trajectory $\boldsymbol{Y}[k] = \left[\, \boldsymbol{y}[k+1]^{\top} \; \ldots \; \boldsymbol{y}[k+N]^{\top} \,\right]^{\top}$ as close as possible to a certain reference trajectory $\boldsymbol{Y}^{\mathrm{r}}[k] = \left[\, \boldsymbol{y}^{\mathrm{r}}[k+1]^{\top} \; \ldots \; \boldsymbol{y}^{\mathrm{r}}[k+N]^{\top} \,\right]^{\top}$. The system output trajectory $\boldsymbol{Y}[k]$ is a prediction depending on the actual system state $\boldsymbol{x}[k]$ and the optimal input trajectory $\boldsymbol{U}^{\star}[k]$. Once the optimal input trajectory is available after solving the respective open-loop optimal control problem, the first value $\boldsymbol{u}^{\star}[k]$ is applied to the plant, whilst the other values of $\boldsymbol{U}^{\star}[k]$ are disregarded. At the next sampling instant, the whole procedure is repeated—this is a shift of the prediction horizon by one sampling interval, which leads to the common term of receding horizon control. The receding horizon strategy allows to react to a change in the reference trajectory and to external disturbances. Figure 2.5 shows open-loop trajectories, which represent future signals that have not been realized yet, whereas the closed-loop trajectories lie in the past.

MPC is often compared to car driving (Camacho and Bordons, 2004). This analogy shall be recalled here to further clarify the basic idea of MPC. The car characteristics have been gained through driving experience, which can be seen as a kind of system identification. A driver of a car has a desired reference trajectory for a finite time horizon (prediction horizon) in his or her mind. For car driving, this prediction horizon is related to the range of vision. Constraints, for example, especially on the deceleration should be considered by the driver, as the car is not able to stop instantly. By taking into account the specific characteristics of the car, the driver determines a sequence of future control actions that allow him or her to follow the reference trajectory as close as possible. The first control action—a combination of acceleration/deceleration, and steering—is then realized by the driver. The rest of the control actions in the mind of the driver are disregarded. The whole procedure is then repeated in a receding horizon fashion in order to be able to react to a changing driving situation, which might be seen as a disturbance.

2.2.2 Preliminaries

An essential part of a successful application of MPC is a precise model of the plant that is controlled. This model might be gathered through physical modeling with subsequent parametrization (grey box identification) or through black box identification where prior knowledge about the system is *not* available (Ljung, 1999).

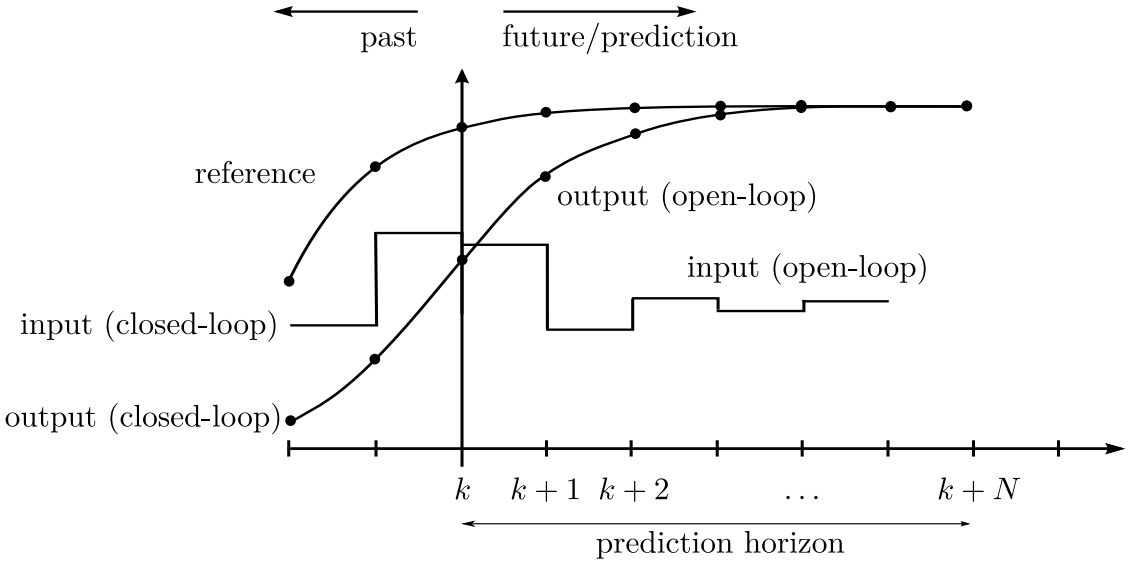


Figure 2.5: The principle of MPC for a system with one input and one output. The future input is determined to steer the predicted output to the reference. Predicted/future values are open-loop values; the closed-loop values are in the past.

Classical MPC applications rely on a linear time-invariant (LTI) discrete-time model in state-space representation (Maciejowski, 2002; Camacho and Bordons, 2004) in the form of

$$\begin{aligned} \boldsymbol{x}[k+1] &= \boldsymbol{A}\boldsymbol{x}[k] + \boldsymbol{B}\boldsymbol{u}[k], \\ \boldsymbol{y}[k] &= \boldsymbol{C}\boldsymbol{x}[k], \end{aligned} \tag{2.12}$$

where $\boldsymbol{x}[k] \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{u}[k] \in \mathbb{R}^{n_u}$ is the system input vector, and $\boldsymbol{y}[k] \in \mathbb{R}^{n_y}$ is the output vector. The numbers of system states, inputs, and outputs are n_x, n_u and n_y, respectively. The matrices $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{B} \in \mathbb{R}^{n_x \times n_u}$, and $\boldsymbol{C} \in \mathbb{R}^{n_y \times n_x}$ are the state matrix, the input matrix, and the output matrix, respectively. The formulation of (2.12) in matrix-vector form describes a multiple-input, multiple-output (MIMO) system. When dealing with single-input, single-output (SISO) systems in this work, the notation of (2.12) is maintained, however, the number of inputs n_u reduces to $n_u = 1$ and the number of outputs n_y reduces to $n_y = 1$.

LTI state-space models often contain $\boldsymbol{y}[k] = \boldsymbol{C}\boldsymbol{x}[k] + \boldsymbol{D}\boldsymbol{u}[k]$ as their output equation, where the matrix $\boldsymbol{D} \in \mathbb{R}^{n_y \times n_u}$ is a so-called feed-through matrix which means that the input $\boldsymbol{u}[k]$ can directly influence the output $\boldsymbol{y}[k]$ at the same time instant. Practical discrete-time control systems always introduce a time-delay between the measurement of $\boldsymbol{y}[k]$ and the time where $\boldsymbol{u}[k]$ is applied to the plant, because the control algorithm needs some time to be calculated (computational delay). Furthermore, actuators, for example a pulse width modulation (PWM) stage, always introduce some delay. Hence, there is no direct feed-through in practice (Maciejowski, 2002, p. 37), which is the reason why $\boldsymbol{D}$ is omitted here.

Linear time-invariant (LTI) systems in the form of (2.12) are governed by *linear* difference equations. *Time-invariant* means that the system matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ do not vary with time.

MPC relies on the availability of the full state vector $\boldsymbol{x}[k]$ for all k. If the full state vector is not available, the plant needs to be observable (Franklin et al., 1997, p. 345ff.) by fulfilling

$$\operatorname{rank}\left(\begin{bmatrix}\boldsymbol{C}^\top & \boldsymbol{C}\boldsymbol{A}^\top & \dots & \boldsymbol{C}\boldsymbol{A}^{n_x-1^\top}\end{bmatrix}^\top\right) = n_x \tag{2.13}$$

in order to use a state estimator like a Kalman filter (Kalman, 1960) or a Luenberger observer (Luenberger, 1964) to estimate the actual state $\boldsymbol{x}[k]$ through measurement(s).

MPC takes the future behavior of the system into account. The trajectories of future states $\boldsymbol{X}[k]$, inputs $\boldsymbol{U}[k]$, outputs $\boldsymbol{Y}[k]$, and reference values $\boldsymbol{Y}^\mathrm{r}[k]$ are written as the stacked vectors

$$\begin{aligned}
\boldsymbol{X}[k] &= \begin{bmatrix}\boldsymbol{x}[k+1]^\top & \boldsymbol{x}[k+2]^\top & \dots & \boldsymbol{x}[k+N]^\top\end{bmatrix}^\top \in \mathbb{R}^{N\cdot n_x},\\
\boldsymbol{U}[k] &= \begin{bmatrix}\boldsymbol{u}[k]^\top & \boldsymbol{u}[k+1]^\top & \dots & \boldsymbol{u}[k+N-1]^\top\end{bmatrix}^\top \in \mathbb{R}^{N\cdot n_u},\\
\boldsymbol{Y}[k] &= \begin{bmatrix}\boldsymbol{y}[k+1]^\top & \boldsymbol{y}[k+2]^\top & \dots & \boldsymbol{y}[k+N]^\top\end{bmatrix}^\top \in \mathbb{R}^{N\cdot n_y},\\
\boldsymbol{Y}^\mathrm{r}[k] &= \begin{bmatrix}\boldsymbol{y}^\mathrm{r}[k+1]^\top & \boldsymbol{y}^\mathrm{r}[k+2]^\top & \dots & \boldsymbol{y}^\mathrm{r}[k+N]^\top\end{bmatrix}^\top \in \mathbb{R}^{N\cdot n_y},
\end{aligned} \tag{2.14}$$

where N is the prediction horizon. The reference vector $\boldsymbol{Y}^\mathrm{r}[k]$ consists of $\boldsymbol{y}^\mathrm{r}[k+1] = \boldsymbol{y}^\mathrm{r}[k+2] = \ldots = \boldsymbol{y}^\mathrm{r}[k+N]$ if the reference is not known in advance. The terms *preview* (used in this work), *look-ahead*, or *anticipative-action* describe the case where the reference is known in advance.

The predicted states[2] $\boldsymbol{X}[k]$ and the predicted outputs $\boldsymbol{Y}[k]$ can be calculated as

$$\begin{aligned}
\boldsymbol{X}[k] &= \mathcal{A}_x\boldsymbol{x}[k] + \mathcal{B}_x\boldsymbol{U}[k],\\
\boldsymbol{Y}[k] &= \mathcal{A}_y\boldsymbol{x}[k] + \mathcal{B}_y\boldsymbol{U}[k],
\end{aligned} \tag{2.15}$$

depending on the future inputs $\boldsymbol{U}[k]$ and the actual state $\boldsymbol{x}[k]$ (Camacho and Bordons, 2004, p. 29). The future input vector $\boldsymbol{U}[k]$ is the optimization variable, which is determined by an

[2]The predicted states are denoted by $\boldsymbol{X}[k]$ and *not* $\boldsymbol{X}[k+1]$ because $\boldsymbol{X}[k]$ is already defined with its first element being $\boldsymbol{x}[k+1]$ (see (2.14)); the same holds for $\boldsymbol{Y}[k]$ and $\boldsymbol{Y}^\mathrm{r}[k]$

optimization problem solver. The prediction matrices for the states, $\mathcal{A}_x$ and $\mathcal{B}_x$, are defined as

$$\begin{aligned}\mathcal{A}_x &= \begin{bmatrix}\boldsymbol{A}^1 & \boldsymbol{A}^2 & \dots & \boldsymbol{A}^N\end{bmatrix}^\top \in \mathbb{R}^{N\cdot n_x \times n_x},\\ \mathcal{B}_x &= \begin{bmatrix}\boldsymbol{A}^0\boldsymbol{B} & \boldsymbol{0}_{n_x\times n_u} & \dots & \dots & \boldsymbol{0}_{n_x\times n_u}\\ \boldsymbol{A}^1\boldsymbol{B} & \boldsymbol{A}^0\boldsymbol{B} & \ddots & & \vdots\\ \vdots & \ddots & \ddots & \ddots & \vdots\\ \vdots & & \ddots & \ddots & \boldsymbol{0}_{n_x\times n_u}\\ \boldsymbol{A}^{N-1}\boldsymbol{B} & \boldsymbol{A}^{N-2}\boldsymbol{B} & \dots & \dots & \boldsymbol{A}^0\boldsymbol{B}\end{bmatrix} \in \mathbb{R}^{N\cdot n_x \times N\cdot n_u}.\end{aligned} \tag{2.16}$$

The prediction matrices for the outputs, $\mathcal{A}_y$ and $\mathcal{B}_y$, are

$$\begin{aligned}\mathcal{A}_y &= \left(\oplus_{i=1}^{N}\boldsymbol{C}\right)\mathcal{A}_x \in \mathbb{R}^{N\cdot n_y \times n_x},\\ \mathcal{B}_y &= \left(\oplus_{i=1}^{N}\boldsymbol{C}\right)\mathcal{B}_x \in \mathbb{R}^{N\cdot n_y \times N\cdot n_u}.\end{aligned} \tag{2.17}$$

The rate of input change $\boldsymbol{\Delta U}[k]$, which is often used for the MPC problem formulation, is defined as

$$\boldsymbol{\Delta U}[k] = \begin{bmatrix}\boldsymbol{u}[k+1]-\boldsymbol{u}[k]\\ \boldsymbol{u}[k+2]-\boldsymbol{u}[k+1]\\ \vdots\\ \boldsymbol{u}[k+N-1]-\boldsymbol{u}[k+N-2]\\ \boldsymbol{u}[k+N-1]\end{bmatrix} \in \mathbb{R}^{N\cdot n_u}, \tag{2.18}$$

and can be written in matrix-vector notation depending on the optimization variable $\boldsymbol{U}[k]$ as

$$\boldsymbol{\Delta U}[k] = \mathcal{B}_\Delta \boldsymbol{U}[k], \tag{2.19}$$

with

$$\mathcal{B}_\Delta = \begin{bmatrix}-\boldsymbol{I}_{n_u} & \boldsymbol{I}_{n_u} & \boldsymbol{0}_{n_u\times n_u} & \dots & \dots & \boldsymbol{0}_{n_u\times n_u}\\ \boldsymbol{0}_{n_u\times n_u} & -\boldsymbol{I}_{n_u} & \boldsymbol{I}_{n_u} & \ddots & & \vdots\\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots\\ \vdots & & \ddots & \ddots & \ddots & \boldsymbol{0}_{n_u\times n_u}\\ \vdots & & & \ddots & -\boldsymbol{I}_{n_u} & \boldsymbol{I}_{n_u}\\ \boldsymbol{0}_{n_u\times n_u} & \dots & \dots & \dots & \boldsymbol{0}_{n_u\times n_u} & \boldsymbol{I}_{n_u}\end{bmatrix} \in \mathbb{R}^{N\cdot n_u \times N\cdot n_u}. \tag{2.20}$$

Choosing $\boldsymbol{u}[k+N-1]$—and not, for example, zero—as the last entry of $\boldsymbol{\Delta U}[k]$ leads to a full rank matrix $\mathcal{B}_\Delta$. This fact, which is discussed in detail in Section 2.2.3, is important to ensure convexity of the cost function.

The handling of constraints is another key part of MPC problems. Symmetric constraints on the input are expressed as

$$|\boldsymbol{U}[k]| \leq \boldsymbol{U}_{\max}, \tag{2.21}$$

where

$$\boldsymbol{U}_{\max} = \begin{bmatrix} \boldsymbol{u}_{\max}^\top & \boldsymbol{u}_{\max}^\top & \dots & \boldsymbol{u}_{\max}^\top \end{bmatrix}^\top \in \mathbb{R}^{N \cdot n_u}. \tag{2.22}$$

The maximum input vector $\boldsymbol{u}_{\max} \in \mathbb{R}^{n_u}$ contains the maximum input values of the n_u inputs. Symmetric constraints on the states are defined using (2.16) as

$$\underbrace{|\mathcal{A}_x \boldsymbol{x}[k] + \mathcal{B}_x \boldsymbol{U}[k]|}_{|\boldsymbol{X}[k]|} \leq \boldsymbol{X}_{\max}, \tag{2.23}$$

where

$$\boldsymbol{X}_{\max} = \begin{bmatrix} \boldsymbol{x}_{\max}^\top & \boldsymbol{x}_{\max}^\top & \dots & \boldsymbol{x}_{\max}^\top \end{bmatrix}^\top \in \mathbb{R}^{N \cdot n_x}. \tag{2.24}$$

The maximum state vector $\boldsymbol{x}_{\max} \in \mathbb{R}^{n_x}$ contains the maximum state values of the n_x states.

Expressing the input and state constraints as component-wise linear inequalities leads to

$$\mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k], \tag{2.25}$$

with

$$\begin{aligned} \mathcal{G} &= \begin{bmatrix} \boldsymbol{I}_{N \cdot n_u \times N \cdot n_u} \\ -\boldsymbol{I}_{N \cdot n_u \times N \cdot n_u} \\ \mathcal{B}_x \\ -\mathcal{B}_x \end{bmatrix} \in \mathbb{R}^{2 \cdot N \cdot (n_u + n_x) \times N \cdot n_u}, \\ \mathcal{H}[k] &= \begin{bmatrix} \boldsymbol{U}_{\max} \\ \boldsymbol{U}_{\max} \\ \boldsymbol{X}_{\max} - \mathcal{A}_x \boldsymbol{x}[k] \\ \boldsymbol{X}_{\max} + \mathcal{A}_x \boldsymbol{x}[k] \end{bmatrix} \in \mathbb{R}^{2 \cdot N \cdot (n_u + n_x)}. \end{aligned} \tag{2.26}$$

The matrices $\mathcal{G}$ and $\mathcal{H}[k]$ are valid if constraints on *every* input and state are imposed, which leads to $2 \cdot N \cdot (n_u + n_x)$ constraints. Other configurations can easily be achieved by deleting the rows corresponding to unconstrained inputs and states in $\mathcal{G}$ and $\mathcal{H}[k]$ leading to a reduced number of constraints. The constraints that are considered in this thesis represent symmetric limitations; however, it is also possible to use maximum and minimum values that have different absolute values. Furthermore, the constraints $\boldsymbol{U}_{\max}$ and $\boldsymbol{X}_{\max}$ are allowed to be time dependent ($\boldsymbol{U}_{\max}[k]$ and $\boldsymbol{X}_{\max}[k]$).

2.2.3 Model Predictive Control Optimization Problem Formulation

One of the most essential steps in MPC is the formulation of the underlying optimization problem. This optimization problem consists of a cost function that should be minimized and constraints that should simultaneously be respected.

A cost function based on a quadratic form is the de facto standard in MPC formulations, because this type of cost function can be cast as a quadratic programming problem for which reliable and efficient solvers are available.

The future control error $\boldsymbol{E}^{\mathrm{r}}[k]$ is a key component of cost functions used in MPC and is defined as

$$\boldsymbol{E}^{\mathrm{r}}[k] = \boldsymbol{Y}[k] - \boldsymbol{Y}^{\mathrm{r}}[k] \in \mathbb{R}^{N \cdot n_y}. \tag{2.27}$$

A popular cost function, which is also used in this work, is given as (Maciejowski, 2002; Camacho and Bordons, 2004)

$$J_{\mathrm{q}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \, \boldsymbol{E}^{\mathrm{r}}[k] \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}} \, \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2. \tag{2.28}$$

This cost function has multiple objectives: The future control error $\boldsymbol{E}^{\mathrm{r}}[k]$ is weighted with the diagonal matrix $\boldsymbol{Q}_{\mathrm{q}}$ and the second objective, the rate of input change $\boldsymbol{\Delta U}[k]$, is weighted with the diagonal matrix $\boldsymbol{R}_{\mathrm{q}}$. Hence, the resulting control performance is a trade-off between the minimization of the future control error $\boldsymbol{E}^{\mathrm{r}}[k]$ and the minimization of $\boldsymbol{\Delta U}[k]$ depending on the chosen weights. Incorporating the rate of input change $\boldsymbol{\Delta U}[k]$ is a common approach to directly influence the shape of the input, for instance to attenuate high-frequency content in the input. Other common forms of cost functions include $\boldsymbol{U}[k]$ instead of $\boldsymbol{\Delta U}[k]$; the difference between both choices is discussed in detail later (see Section 5.2). The close relationship of the cost function (2.28) to MPC with integral action is shown in (Maciejowski, 2002, p. 49).

The weighting matrices are defined as

$$\begin{aligned} \boldsymbol{Q}_{\mathrm{q}} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{q}_{\mathrm{q}}\} \in \mathbb{R}^{N\cdot n_y \times N\cdot n_y}, \\ \boldsymbol{R}_{\mathrm{q}} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\mathrm{q}}\} \in \mathbb{R}^{N\cdot n_u \times N\cdot n_u}, \end{aligned} \tag{2.29}$$

where $\boldsymbol{q}_{\mathrm{q}} \in \mathbb{R}_{+}^{n_y}$ and $\boldsymbol{r}_{\mathrm{q}} \in \mathbb{R}_{+}^{n_u}$ are the weights for the control error and the rate of input change, respectively. The weighting matrices $\boldsymbol{Q}_{\mathrm{q}}$ and $\boldsymbol{R}_{\mathrm{q}}$ are forced to be positive definite ($\boldsymbol{Q}_{\mathrm{q}} \succ 0$ and $\boldsymbol{R}_{\mathrm{q}} \succ 0$) by choosing every element of $\boldsymbol{q}_{\mathrm{q}}$ and $\boldsymbol{r}_{\mathrm{q}}$ positive. This choice leads to a convex cost function.

The cost function (2.28) is expressed, depending on the optimization variable $\boldsymbol{U}[k]$, as

$$J_{\mathrm{q}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \overset{\substack{(2.17)\\(2.19)}}{=} \left\| \underbrace{\begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \mathcal{B}_y \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}} \mathcal{B}_{\Delta} \end{bmatrix}}_{\boldsymbol{F}_{\mathrm{q}}} \boldsymbol{U}[k] + \underbrace{\begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \left(\mathcal{A}_y \boldsymbol{x}[k] - \boldsymbol{Y}^{\mathrm{r}}[k]\right) \\ \boldsymbol{0}_{N\cdot n_u} \end{bmatrix}}_{\boldsymbol{g}_{\mathrm{q}}[k]} \right\|_2^2 . \tag{2.30}$$

Applying $\|\boldsymbol{a}\|_2^2 = \boldsymbol{a}^\top \boldsymbol{a}$ to (2.30) leads to the quadratic form

$$J_{\mathrm{q}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \boldsymbol{U}[k]^\top \boldsymbol{F}_{\mathrm{q}}^\top \boldsymbol{F}_{\mathrm{q}} \boldsymbol{U}[k] + 2\boldsymbol{g}_{\mathrm{q}}[k]^\top \boldsymbol{F}_{\mathrm{q}} \boldsymbol{U}[k] + \boldsymbol{g}_{\mathrm{q}}[k]^\top \boldsymbol{g}_{\mathrm{q}}[k]. \tag{2.31}$$

By combining the cost function (2.31) with the constraints (2.25), the final MPC optimization problem is

$$\begin{aligned} \underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \boldsymbol{U}[k]^\top \boldsymbol{F}_{\mathrm{q}}^\top \boldsymbol{F}_{\mathrm{q}} \boldsymbol{U}[k] + 2\boldsymbol{g}_{\mathrm{q}}[k]^\top \boldsymbol{F}_{\mathrm{q}} \boldsymbol{U}[k] + \boldsymbol{g}_{\mathrm{q}}[k]^\top \boldsymbol{g}_{\mathrm{q}}[k] \\ \text{subject to} \quad & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{2.32}$$

This QP in standard form (see (2.5)) represents the key part of MPC based on quadratic forms—it is solved at every sampling instant while the first entry of the optimal input vector $\boldsymbol{U}^\star[k]$, namely $\boldsymbol{u}^\star[k]$, is applied to the plant.

Neglecting the inequality constraints $\mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]$ in (2.32) would allow deriving the analytic solution

$$\boldsymbol{U}^\star[k] = -\left(\boldsymbol{F}_{\mathrm{q}}^\top \boldsymbol{F}_{\mathrm{q}}\right)^{-1} \left(\boldsymbol{F}_{\mathrm{q}}^\top \boldsymbol{g}_{\mathrm{q}}[k]\right), \tag{2.33}$$

which is the solution of an unconstrained finite horizon optimal control problem.

A required property of a QP is its convexity, as it determines if the optimal input vector $\boldsymbol{U}^\star[k]$ represents a global minimum. Following the reasoning of Section 2.1.3, the constraints of (2.32) are affine and therefore form a convex set. It remains to show that the cost

function of (2.32) is convex, which is equivalent to the positive definiteness of the quadratic term

$$\boldsymbol{F}_{\mathrm{q}}^{\top}\boldsymbol{F}_{\mathrm{q}} = \mathcal{B}_y^{\top}\underbrace{\boldsymbol{Q}_{\mathrm{q}}}_{\succ 0}\mathcal{B}_y + \mathcal{B}_{\Delta}^{\top}\underbrace{\boldsymbol{R}_{\mathrm{q}}}_{\succ 0}\mathcal{B}_{\Delta} \in \mathbb{R}^{N\cdot n_u \times N\cdot n_u}. \tag{2.34}$$

A positive definite weight $\boldsymbol{Q}_{\mathrm{q}} \succ 0$ leads to $\mathcal{B}_y^{\top}\boldsymbol{Q}_{\mathrm{q}}\mathcal{B}_y \succeq 0$. As $\mathcal{B}_{\Delta}$ has full rank, it follows that $\mathcal{B}_{\Delta}^{\top}\boldsymbol{R}_{\mathrm{q}}\mathcal{B}_{\Delta} \succ 0$ which leads to the final result $\boldsymbol{F}_{\mathrm{q}}^{\top}\boldsymbol{F}_{\mathrm{q}} \succ 0$. Hence, the optimization problem (2.32) is convex. Convexity simultaneously leads to the existence of the inverse in (2.33).

2.2.4 Feasibility

If no solution to an optimization problem exists, the problem is infeasible. A problem is feasible if the cost function is bounded and all constraints can be satisfied. Reasons for infeasibility might be unobtainable control objectives and conflicting constraints (Camacho and Bordons, 2004, p. 197). Unobtainable control objectives can be prevented by a careful choice of reference values. For example, if constraints on the manipulated variable need to be considered, reference values should be chosen that do not violate these constraints. A common approach to improve feasibility is the proper management of constraints. It is important to distinguish between input and state constraints. State constraints can cause infeasibility, for example, in the presence of a disturbance or a model mismatch when the actual state (gained through measurement or estimation) lies outside the state constraints. Input constraints are usually not that critical, as the input is resulting from the optimization problem and is therefore, for example, not directly influenced by a disturbance. It is common to introduce hard and soft constraints. Examples for hard constraints are critical limitations that must not be violated and actuator limitations that cannot be violated. Soft constraints are limitations that are allowed to be exceeded (normally for a short time); they are usually realized using so-called slack variables (Camacho and Bordons, 2004, pp. 198–199). A drawback of slack variables is the increased optimization problem size, which usually leads to a prolonged solution time (Maciejowski, 2002, p. 101). Input constraints are usually considered hard, whereas state constraints are considered soft. Another possibility to improve the feasibility concerning constraints is to remove the state constraints for the first few time instants, where constraint infeasibility often occurs because of sudden perturbations or measurement noise (Camacho and Bordons, 2004, p. 199).

2.2.5 Stability

Stability is an elementary issue in control systems engineering, only stable closed-loop systems are applicable in practice and allow safe operation.

- A system is *stable* if the unforced response (input is zero) of a system approaches zero as time approaches infinity.
- A system is *unstable* if the unforced response gets unbounded as time approaches infinity.
- A system is *marginally stable* if the unforced response remains constant or oscillates with a constant amplitude as time approaches infinity (Nise, 2011, p. 302).

These definitions lead to the following stability conditions for a discrete-time LTI system in the form of (2.12) (Nise, 2011, p. 743):

- *Stable*: All eigenvalues of the state matrix $\boldsymbol{A}$ lie inside the unit-circle.
- *Unstable*: One or more eigenvalues of the state matrix $\boldsymbol{A}$ lie outside the unit-circle. Furthermore, if an eigenvalue of multiplicity greater than one is located on the unit-circle, the system is unstable[3].
- *Marginally stable*: An eigenvalue of the state matrix $\boldsymbol{A}$ of multiplicity one lies on the unit-circle and all other eigenvalues are inside the unit-circle.

These stability conditions are valid as long as constraints do not get active. As MPC is a control method that inherently handles constraints, the closed-loop system is *nonlinear* even if the plant to be controlled is *linear*. This complicates the theory of stability for MPC. Saberi et al. (2000) pointed out that an unstable LTI system with input constraints cannot be *globally* stabilized; statements of stability for MPC are therefore *always local* in the case of an unstable system. The feasibility of an optimization problem is a prerequisite for stability considerations of MPC.

Mayne et al. (2000) consider the stability theory of MPC in their paper (with more than 3800 Google Scholar™ citations) to be at a "relatively mature stage". However, the majority of stability proofs deal with *regulation* problems whose goal is to regulate the states to the origin. Nevertheless, a system can be transformed such that the origin represents a *constant* reference (Mayne et al., 2000). The fact that MPC of constrained systems represents a nonlinear control approach necessitates the use of the stability theory of Lyapunov originating from his dissertation in 1892 (Lyapunov, 1992, English reprint). Most stability

[3] A simple double integrator has two eigenvalues on the unit-circle and is therefore unstable.

proofs employ the cost function as a Lyapunov function. Stabilizing MPC is gained through modifications of the optimization problem. Well-known modifications of the MPC problem for unstable systems are the use of terminal constraints and a terminal cost. If the system that is controlled is stable, terminal constraints can be omitted (Mayne et al., 2000). The design of a stable MPC scheme for an unstable plant is therefore more complex than the design for a stable plant. The basic idea behind the modifications of the optimization problem is to emulate infinite horizon control, with its advantages of stability and robustness. Cairano and Bemporad (2010) presented an MPC design method based on controller matching, which means that the MPC optimization problem is designed such that the control results correspond to a given linear controller, for example a linear quadratic regulator (LQR), if the constraints are not active. The problem of stability was not directly addressed in industry; engineers often restricted the applications to stable plants and chose a large prediction horizon compared to the plant settling time—these measures also emulate infinite horizon control (Mayne et al., 2000). Stability issues and measures for direct MPC, where the input variable belongs to a finite control set, are discussed in (Aguilera and Quevedo, 2011).

The discussed modifications to the optimization problem allow to proof stability for MPC *regulation* problems. In order to deal with *arbitrary* varying reference signals, the concept of reference governors (RGs) was developed simultaneously by Gilbert and Tan (1995) and Bemporad and Mosca (1995). An RG allows to decouple—in contrast to MPC—the problems of stabilizing a possibly unstable plant and the handling of constraints. It is assumed that the plant, in absence of constraints, is equipped with a stabilizing controller with good tracking behavior. The handling of constraints is the task of the RG, which modifies the reference signal in order to prevent the plant inputs and states from saturating.

2.2.6 Tuning of the Cost Function Weights

The concept of MPC is often described as “intuitively appealing” (Maciejowski, 2002, p. 31) or as “intuitive and easy to understand” (Cortes et al., 2008). These statements might be true for the basic concept of MPC, but they are also often generalized for the tuning of the respective cost function weights. However, when it comes to weights tuning, only “rules of thumb” are available most of the time—especially for the multivariable case of MIMO systems or for the case where multiple objectives, which might even be conflicting, are included in a single cost function (Maciejowski, 2002, p. 188).

The tuning of weights influences the control performance. Hackl et al. (2013) illustrated in their work by a simple third order system (position control) that the tuning of cost function weights might be *not intuitive*; even for this simple example, the control performance is

shown to be a non-convex function of the weights. This means in this case that increasing the control error weight does not necessary lead to better control performance, as the control performance contour shows multiple local minima.

Garriga and Soroush (2010) collected various MPC tuning methods in their review; the sheer number of methods in their work illustrates that the tuning of weights is not straightforward and is still an open research topic. Hackl et al. (2013) suggested small changes of the weighting matrices that can make tuning more intuitive. Other researchers aim to ease the process of weight tuning through automatic tuning methods (Garriga and Soroush, 2010; Waschl et al., 2012). Guidelines for weighting factor design for direct MPC are given in (Cortes et al., 2009).

The critical reflection about the tuning of weights should *not* discourage the application of MPC; it should rather strengthen the awareness of this often underestimated topic. For applications that require a fast dynamic response and simultaneous handling of constraints, it might be challenging or even impossible to tune traditional linear controllers to meet both requirements.

2.2.7 Choice of the Prediction Horizon

Besides the weighting matrices in the cost function, the prediction horizon N is another important parameter in the design of MPC laws. A long prediction horizon increases the problem size and therefore the time that is required to solve the optimization problem. It is therefore necessary to find a trade-off between computation time and stability considerations. The influence of the prediction horizon on the control result can be illustrated by comparing open-loop predictions with the closed-loop trajectories for a perfectly known plant. An important proposition drawn from Bellman's principle of optimality (Bellman, 1956) is that for an infinite prediction horizon, open-loop and closed-loop trajectories coincide (Maciejowski, 2002, p. 173). As discussed in Section 2.2.5 an infinite prediction horizon is often desired to achieve a stable closed-loop system (Mayne et al., 2000).

Figure 2.6 illustrates possible consequences of a short prediction horizon and a long prediction horizon. The MPC scheme with the short prediction horizon leads to a fast increasing output. However, this could lead to a large overshoot, as the horizon is too short for the controller to "see" that it is approaching the reference too fast. Consequently, the open- and closed-loop trajectories differ. A large enough prediction horizon covers the whole time span in which the reference is reached. The open- and closed-loop trajectories coincide because a long prediction horizon emulates an infinite horizon.

From another point of view, open-loop and closed-loop trajectories that do not coincide lead to suboptimal control results. *Suboptimal* in this context means that the value of the

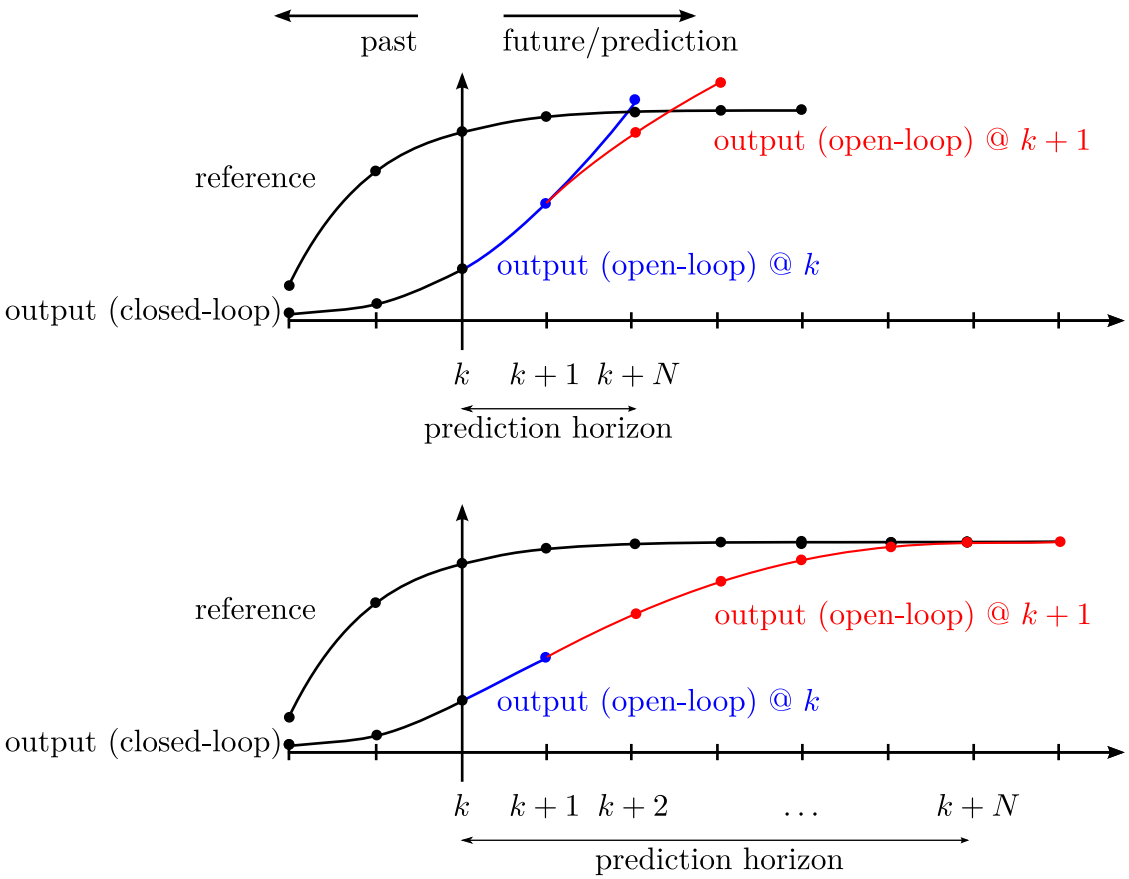


Figure 2.6: Choice of the prediction horizon: The top figure shows two open-loop trajectories of the output for a short prediction horizon—as the open-loop predictions of the output computed at k and $k+1$ do not coincide, the open- and closed-loop trajectories will also differ. The behavior for a long (enough) prediction horizon is shown in the bottom figure: The predicted output trajectories at k and $k+1$ coincide, which will lead to coinciding open- and closed-loop trajectories.

cost function calculated for the closed-loop response is higher than the cost of the open-loop response (Stumper, 2013, p. 91).

Summing up, it is advisable to choose a prediction horizon that is higher than the settling time of the plant in order to achieve stability properties equivalent to infinite horizon control and to get optimal trajectories with respect to the chosen cost function.

2.2.8 Numerical Aspects

A crucial task when dealing with optimization problems is to provide a well conditioned problem to the respective optimization problem solver. The numerical conditioning of a problem can change the convergence rate and termination tests—hence, a badly scaled problem can make a good optimization algorithm bad (Betts, 2001, p. 35). The convergence rate and termination tests in turn influence the computation time of the respective optimization algorithm.

The numerical conditioning of an optimization problem is influenced by:

- The tuning of the cost function weights $\boldsymbol{Q}_{\mathrm{q}}$ and $\boldsymbol{R}_{\mathrm{q}}$: The tuning is to a large extent "fixed" because of dynamic requirements. However, setting a weight much larger or smaller than necessary deteriorates the numerical conditioning.

- The prediction horizon: Mainly determined through stability and optimality considerations (see Sections 2.2.5 and 2.2.7), the prediction horizon N has a large influence on the conditioning of the optimization problem, because N appears as an exponent in the prediction matrices (2.17). It is therefore reasonable to keep the prediction horizon as short as possible while still fulfilling stability and optimality requirements. As discussed in Section 2.2.7, the prediction horizon is often chosen to be slightly higher than the settling time of the plant.

- The conditioning of the system model: In contrast to the first two points, the conditioning of the system model can be improved without having a large influence on the overall control performance. Unfortunately, due to the interaction with a solver, there are no straight rules to get a well-scaled optimization problem. Still, a first step often is the scaling of the system states and inputs such that they are all in the same order of magnitude (Betts, 2001, p. 35).

In order to improve the numerical conditioning of the optimization problems of this work, scaling of system states and system inputs is applied through the following transformation (Franklin et al., 2002, p. 74)

$$\begin{aligned} \tilde{\boldsymbol{x}}[k] &= \boldsymbol{S}_x^{-1}\boldsymbol{x}[k], \\ \tilde{\boldsymbol{u}}[k] &= \boldsymbol{S}_u^{-1}\boldsymbol{u}[k], \end{aligned} \tag{2.35}$$

where $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{u}}$ are the scaled state and input vectors, respectively. The matrices $\boldsymbol{S}_x$ and S_u are the corresponding diagonal transformation matrices

$$\begin{aligned} \boldsymbol{S}_x &= \mathrm{diag}\left\{\boldsymbol{x}_{\max}\right\} \in \mathbb{R}^{n_x \times n_x}, \\ \boldsymbol{S}_u &= \mathrm{diag}\left\{\boldsymbol{u}_{\max}\right\} \in \mathbb{R}^{n_u \times n_u}, \end{aligned} \tag{2.36}$$

where $\boldsymbol{x}_{\max}$ and $\boldsymbol{u}_{\max}$ are the maximum (or perhaps nominal) values of the respective states and inputs in order to bring the values of $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{u}}$ to the range of ± 1.

Applying this transformation to an unscaled discrete-time system in the form of (2.12) leads to

$$\begin{aligned} \tilde{\boldsymbol{x}}[k+1] &= \underbrace{\boldsymbol{S}_x^{-1}\boldsymbol{A}\boldsymbol{S}_x}_{\tilde{\boldsymbol{A}}}\tilde{\boldsymbol{x}}[k] + \underbrace{\boldsymbol{S}_x^{-1}\boldsymbol{B}\boldsymbol{S}_u}_{\tilde{\boldsymbol{B}}}\tilde{\boldsymbol{u}}[k], \\ \tilde{\boldsymbol{y}}[k] &= \boldsymbol{C}\tilde{\boldsymbol{x}}[k]. \end{aligned} \tag{2.37}$$

The scaled quantities $\tilde{\boldsymbol{x}}[k]$, $\tilde{\boldsymbol{u}}[k]$, $\tilde{\boldsymbol{y}}[k]$, $\tilde{\boldsymbol{A}}$, and $\tilde{\boldsymbol{B}}$ are used internally in the presented control algorithms. The scaled output $\tilde{\boldsymbol{y}}[k]$ is based on the unchanged output matrix $\boldsymbol{C}$. The whole notation of this work and the experimental results are, however, for the sake of notational simplicity, based on unscaled quantities. Scaling of a continuous-time state-space model can be done analogously to the discrete-time case.

The scaling of the system description is not only advantageous in order to improve the optimization problem conditioning, it can also make the tuning of MPC weighting factors easier (Hackl et al., 2013), as the terms in the cost function are brought to approximately the same numerical range.

Another possibility of scaling is to determine the transformation matrices (2.36) such that approximately equal row and column norms in the system matrices $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{B}}$ are achieved. This so-called balancing can lead to an improved numerical robustness (Parlett and Reinsch, 1969).

Other approaches to improve the numerical conditioning try to scale the optimization problem *after* it has been formulated (in contrast to the scaling of the state-space model). These preconditioning methods aim to improve the conditioning of the Hessian (the quadratic term) of the cost function (Bradley, 2010; Richter et al., 2012). The term *pre* in this case refers to the phase immediately before solving the optimization problem.

2.3 Robust Model Predictive Control

The robustness of a control system against uncertainties is—besides the dynamic behavior and stability—another important topic in the design of a controller. Uncertainties can originate from external disturbances, model mismatch, and unmodeled dynamics (Zhou and Doyle, 1998). In general, control properties (e.g., dynamic behavior, constraint satisfaction, etc.) of a *robust* control system are only marginally influenced by uncertainties. This section focuses on parametric uncertainties resulting from a model mismatch, which is either caused by system parameters that change over time or by an inaccurately parametrized model.

Usually, there are three different objectives when the topic of robustness is discussed (Bemporad and Morari, 1999)

- Robust stability: Stability properties can be guaranteed even for an uncertain system.
- Robust constraint handling: Constraints are maintained even for an uncertain system.
- Robust performance: Certain performance criteria (e.g., dynamic performance) are met even for an uncertain system.

As one of the main points of this work is to exploit the features of MPC to improve the dynamic control performance, the focus lies on *robust performance* here. Nevertheless, the objective of robust performance is only reasonable if stability is maintained in presence of uncertain parameters.

Robust model predictive control (RMPC) based on a min-max formulation was introduced by Campo and Morari (1987) with the intention to minimize the worst-case tracking error for an uncertain system. Further improvements were reported by Kothare et al. (1996) where the approach was extended to state-space models and robust stability was proven. However, first simulations showed that the approach of Kothare et al. (1996) is too conservative to achieve a fast dynamic control performance for the application considered in this work. In this context, *conservative* means that the constraints were not fully exploited, which led to a dynamic behavior that was not satisfactory. Therefore, the approach that is used in this work is based on (Casavola et al., 2000a; Schuurmans and Rossiter, 2000), where so-called enumerative schemes together with a polytopic uncertainty description are used to formulate a robust optimization problem. A min-max formulation based on semidefinite relaxation was introduced by Löfberg (2003) in order to simplify the enumeration based optimization problem, but simulations also showed too conservative control results for this approach.

The basic principle of RMPC is the same as described in Section 2.2.1 for MPC. The comments on feasibility, the choice of the prediction horizon, the tuning of the cost function weights, and the numerical aspects in Section 2.2 also remain unchanged for the RMPC case.

2.3.1 Preliminaries

To deal with parametric uncertainties it is necessary to formulate a system model that includes uncertain parameters. Linear time-varying (LTV) systems in the form of

$$\begin{aligned} \boldsymbol{x}[k+1] &= \boldsymbol{A}[k]\boldsymbol{x}[k] + \boldsymbol{B}[k]\boldsymbol{u}[k], \\ \boldsymbol{y}[k] &= \boldsymbol{C}\boldsymbol{x}[k], \end{aligned} \tag{2.38}$$

can model such parametric uncertainties. The difference to an LTI system is that the system matrices $\boldsymbol{A}[k]$ and $\boldsymbol{B}[k]$ can vary over time.

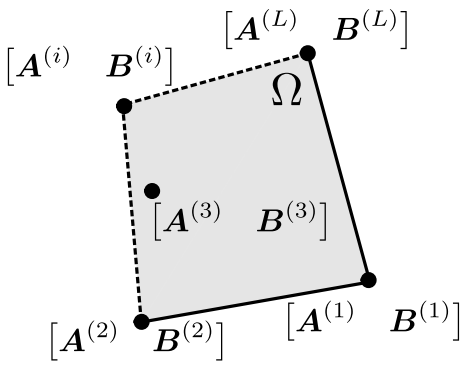


Figure 2.7: Illustration of a polytopic uncertainty model

In order to model the parametric uncertainties in $\boldsymbol{A}[k]$ and $\boldsymbol{B}[k]$, a polytopic uncertainty model is used (Kothare et al., 1996; Wan and Kothare, 2003), where these matrices lie within the polytope Ω

$$\begin{bmatrix}\boldsymbol{A}[k] & \boldsymbol{B}[k]\end{bmatrix} \in \Omega. \tag{2.39}$$

The polytope Ω (see Figure 2.7) is defined as

$$\Omega = \mathrm{Co}\left\{\begin{bmatrix}\boldsymbol{A}^{(1)} & \boldsymbol{B}^{(1)}\end{bmatrix}, \begin{bmatrix}\boldsymbol{A}^{(2)} & \boldsymbol{B}^{(2)}\end{bmatrix}, \ldots, \begin{bmatrix}\boldsymbol{A}^{(L)} & \boldsymbol{B}^{(L)}\end{bmatrix}\right\}, \tag{2.40}$$

which is the convex hull $\mathrm{Co}\{\cdot\}$ of extreme system realizations. The number of extreme realizations L is calculated through combinatorial reasoning as

$$L = 2^{n_\mathrm{p}} \in \mathbb{N}, \tag{2.41}$$

where $n_\mathrm{p} \in \mathbb{N}_0$ is the number of uncertain parameters.

The following example illustrates the concept of a polytopic uncertainty model. For simplicity, a system without an input is considered ($\boldsymbol{B}[k] = \boldsymbol{0}_{n_x \times n_u}$). The time varying matrix

$$\boldsymbol{A}[k] = \begin{bmatrix}1 & 0.1\\ 0 & a_{21}[k]\end{bmatrix} \tag{2.42}$$

serves as an example. The parameter $a_{21}[k]$ is considered to be uncertain in the range of $\pm 10\%$ of its nominal value a_{21}^{nom} (infinitely many values). The polytope Ω of the polytopic uncertainty model is described by *finitely many* extreme realizations

$$\Omega = \mathrm{Co}\left\{\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}\right\}, \tag{2.43}$$

where the extreme realizations $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$ are given by

$$\boldsymbol{A}^{(1)} = \begin{bmatrix} 1 & 0.1 \\ 0 & 1.1 \cdot a_{21}^{\mathrm{nom}} \end{bmatrix}, \; \boldsymbol{A}^{(2)} = \begin{bmatrix} 1 & 0.1 \\ 0 & 0.9 \cdot a_{21}^{\mathrm{nom}} \end{bmatrix}. \tag{2.44}$$

In order to use the polytopic uncertainty model in an RMPC scheme it is necessary to predict the system behavior. However, formulating a robust optimization problem requires—in contrast to the prediction (2.15)—predictions based on an uncertain system. A polytopic uncertainty model with one uncertain parameter ($n_{\mathrm{p}} = 1$) leads to $L = 2^{n_{\mathrm{p}}} = 2$ extreme realizations. Starting with the actual state $\boldsymbol{x}[k]$, the state at the next sampling instant is predicted as

$$\boldsymbol{x}[k+1] = \boldsymbol{A}[k]\boldsymbol{x}[k] \in \mathrm{Co}\Big\{ \underbrace{\boldsymbol{A}^{(1)}\boldsymbol{x}[k]}_{\boldsymbol{x}^{(1)}[k+1]}, \underbrace{\boldsymbol{A}^{(2)}\boldsymbol{x}[k]}_{\boldsymbol{x}^{(2)}[k+1]} \Big\}. \tag{2.45}$$

This means that $\boldsymbol{x}[k+1]$ is not a certain vector but lies within the convex polytope $\mathrm{Co}\left\{\boldsymbol{x}^{(1)}[k+1], \boldsymbol{x}^{(2)}[k+1]\right\}$. In other words, for the exemplary time varying matrix (2.42), *every* prediction with the parameter a_{21} in the range of $\pm 10\%$ of its nominal value a_{21}^{nom} lies within the predictions $\boldsymbol{x}^{(1)}[k+1]$ and $\boldsymbol{x}^{(2)}[k+1]$. It is essential to determine the polytopic uncertainty model such that *all* system realizations of the system that varies over time lie inside the convex hull of the extreme realizations. According to Kothare et al. (1996), a practical approach to determine the polytopic uncertainty model is to identify various discrete-time system models at relevant operating points (e.g., temperatures) and at different times. The polytope Ω is then filled with the identified models.

Predicting the next state $\boldsymbol{x}[k+2]$ using $\boldsymbol{x}[k+1]$ leads to

$$\boldsymbol{x}[k+2] \in \mathrm{Co}\Big\{ \underbrace{\boldsymbol{A}^{(1)}\boldsymbol{x}^{(1)}[k+1]}_{\boldsymbol{x}^{(1)}[k+2]}, \underbrace{\boldsymbol{A}^{(1)}\boldsymbol{x}^{(2)}[k+1]}_{\boldsymbol{x}^{(2)}[k+2]}, \underbrace{\boldsymbol{A}^{(2)}\boldsymbol{x}^{(1)}[k+1]}_{\boldsymbol{x}^{(3)}[k+2]}, \underbrace{\boldsymbol{A}^{(2)}\boldsymbol{x}^{(2)}[k+1]}_{\boldsymbol{x}^{(4)}[k+2]} \Big\}. \tag{2.46}$$

The principle of uncertain prediction is also illustrated in Figure 2.8. This prediction tree illustrates why the later discussed RMPC scheme is called enumerative; finitely many paths through this tree have to be considered. This leads to a high computational complexity.

If the uncertainties are assumed to be time-invariant over the prediction horizon, the prediction tree has $n_{\mathrm{seq}} = L$ possible sequences that lead to uncertain predictions. If the uncertainties are considered to be time-variant over the prediction horizon, $n_{\mathrm{seq}} = L^N$ possible sequences have to be considered (illustrated in (2.46)) which leads to exponential complexity. Time-invariant uncertainties are caused, for example, by inaccurate system identification (parameter mismatch). Further time-invariant parameters are parameters that vary only

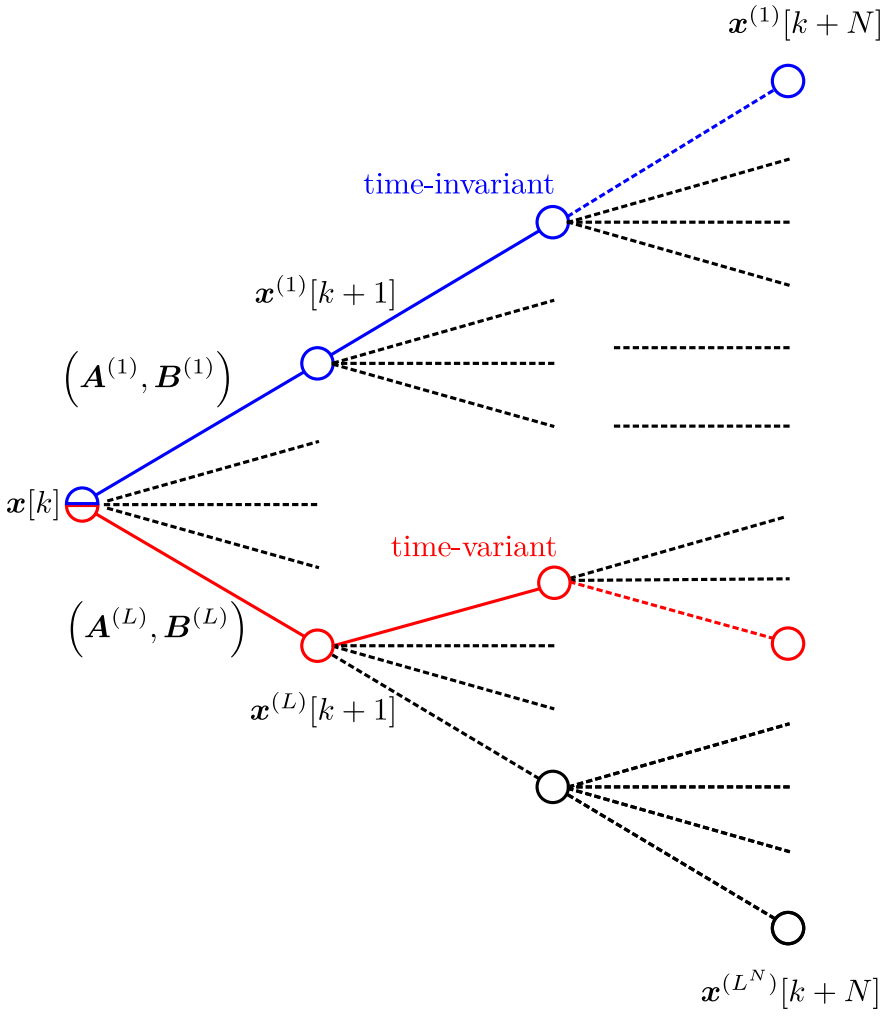


Figure 2.8: Prediction for the polytopic uncertainty model using an enumerative scheme. The time-invariant and the time-variant cases are indicated.

marginally over the prediction horizon (e.g., the resistance of an electric motor that varies slowly with the temperature). The inductance of an electric motor is an example for a parameter that is time-variant over the prediction horizon—magnetic saturation that depends on the electric current can be modeled by a fast changing inductance.

In this work—in order to reduce the computational complexity—the uncertainties are assumed to be time-invariant over the horizon. Speaking in terms of example (2.46), a constant uncertainty leads to $\boldsymbol{x}[k+2] \in \mathrm{Co}\{\boldsymbol{A}^{(1)}\boldsymbol{x}^{(1)}[k+1],\ \boldsymbol{A}^{(2)}\boldsymbol{x}^{(2)}[k+1]\}$.

The l-th uncertain predictions of the states $\boldsymbol{X}^{(l)}[k]$ and outputs $\boldsymbol{Y}^{(l)}[k]$ are written as

$$\begin{aligned} \boldsymbol{X}^{(l)}[k] &= \mathcal{A}_x^{(l)}\boldsymbol{x}[k] + \mathcal{B}_x^{(l)}\boldsymbol{U}[k], \\ \boldsymbol{Y}^{(l)}[k] &= \mathcal{A}_y^{(l)}\boldsymbol{x}[k] + \mathcal{B}_y^{(l)}\boldsymbol{U}[k], \quad \forall l \in \{1, 2, \ldots, n_{\mathrm{seq}}\}. \end{aligned} \tag{2.47}$$

The stacked vectors $\boldsymbol{X}^{(l)}[k]$, $\boldsymbol{Y}^{(l)}[k]$ and $\boldsymbol{U}^{(l)}[k]$ are defined analogously to (2.14). As the uncertainties are considered to be constant over the prediction horizon, the number of se-

quences reads as $n_{\text{seq}} = L = 2^{n_{\text{p}}}$. The interested reader is referred to the work of Schuurmans and Rossiter (2000) for a detailed derivation of $\mathcal{A}_x^{(l)}$, $\mathcal{A}_y^{(l)}$, $\mathcal{B}_x^{(l)}$ and $\mathcal{B}_y^{(l)}$.

The formulation of the rate of input change $\boldsymbol{\Delta U}[k]$ and the constraints on the states $|\boldsymbol{X}| \leq \boldsymbol{X}_{\max}$ and the inputs $|\boldsymbol{U}| \leq \boldsymbol{U}_{\max}$ remain the same as in the MPC preliminaries Section 2.2.2. As the constraints remain the same, robust constraint fulfillment is not considered here. This means that it is not guaranteed that state constraints are fulfilled for the uncertain plant; input constraints are fulfilled, as the formulation of the input constraints does not contain any uncertainties. The robust fulfillment of state constraints can lead to a conservative control behavior (Bemporad and Morari, 1999) and is therefore out of the focus of this work. The choice of non-robust constraints in this work is reasonable, as input constraints are often hard constraints (through actuator limitations) and state constraints are often allowed to be slightly violated. Furthermore, state constraints are usually softened through slack variables and are therefore not exactly fulfilled even when no uncertainties are present (nominal case).

2.3.2 Robust Model Predictive Control Optimization Problem Formulation

The RMPC scheme considered here is based on a so-called min-max formulation introduced by Campo and Morari (1987). The basic idea is to find the worst-case system realization and to minimize the respective worst-case cost function. An enumeration based min-max RMPC scheme, introduced in (Casavola et al., 2000a; Schuurmans and Rossiter, 2000), is presented in detail in the following.

The basic ingredients of the MPC cost function such as the control error and the rate input change remain unchanged in the RMPC formulation. However, the control error is now dependent on uncertain predictions. The cost function for RMPC based on enumeration is written as

$$J_{\text{q,rob}}^{(l)}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\text{r}}[k]) = \left\| \begin{bmatrix} (\boldsymbol{Q}_{\text{q}})^{\frac{1}{2}} \, \boldsymbol{E}^{\text{r},(l)}[k] \\ (\boldsymbol{R}_{\text{q}})^{\frac{1}{2}} \, \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2, \tag{2.48}$$

where $l \in \{1, 2, \ldots, n_{\text{seq}}\}$. This cost function has again two objectives, namely, the future control error $\boldsymbol{E}^{\text{r},(l)}[k]$ and the rate of input change $\boldsymbol{\Delta U}[k]$. The index l identifies the respective path of the prediction tree in Figure 2.8. The weighting matrices $\boldsymbol{R}_{\ell_1}$ and $\boldsymbol{Q}_{\ell_1}$ are defined by (2.29).

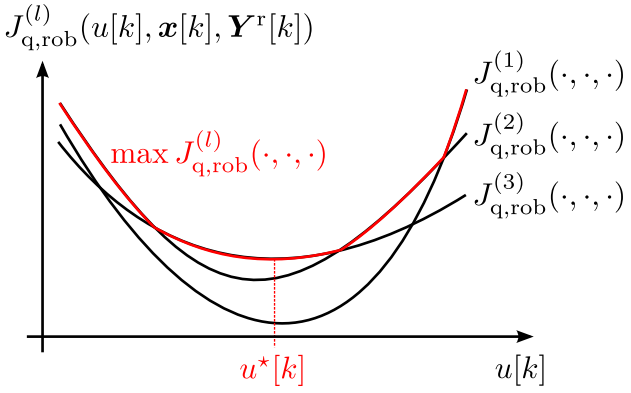


Figure 2.9: Graphical representation of the min-max problem (2.50), with $n_{\mathrm{seq}} = 3$. In this illustration, it is assumed that *all* components of $\boldsymbol{U}[k]$ have the value $u[k]$, which is in turn varied along the abscissa. The optimal value of $u[k]$ is denoted by $u^\star[k]$.

The cost function (2.48) is expressed, depending on the optimization variable $\boldsymbol{U}[k]$, as

$$J^{(l)}_{\mathrm{q,rob}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \overset{\substack{(2.47)\\(2.19)}}{=} \left\| \underbrace{\begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \mathcal{B}_y^{(l)} \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}} \mathcal{B}_\Delta \end{bmatrix}}_{\boldsymbol{F}_{\mathrm{q}}^{(l)}} \boldsymbol{U}[k] + \underbrace{\begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \left(\mathcal{A}_y^{(l)} \boldsymbol{x}[k] - \boldsymbol{Y}^{\mathrm{r}}[k] \right) \\ \boldsymbol{0}_{N \cdot n_u} \end{bmatrix}}_{\boldsymbol{g}_{\mathrm{q}}^{(l)}[k]} \right\|_2^2, \tag{2.49}$$

The min-max based robust optimization problem including the cost function (2.49) and the constraints (2.25), which are already known from the MPC formulation, is stated as

$$\begin{aligned} &\underset{\boldsymbol{U}[k]}{\text{minimize}} && \max_{l \in \{1,2,\ldots,n_{\mathrm{seq}}\}} J^{(l)}_{\mathrm{q,rob}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \\ &\text{subject to} && \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{2.50}$$

The *max* part of the cost function (2.50) represents the worst-case cost function realization. This worst-case realization is minimized over the future control actions $\boldsymbol{U}[k]$. Figure 2.9 illustrates the idea of the min-max formulation by a simple scalar example. What remains, is to bring the optimization problem (2.50) to a standard form. Applying the epigraph transformation of Section 2.1.5 to (2.50) leads to

$$\begin{aligned} &\underset{\boldsymbol{U}[k],\gamma}{\text{minimize}} && \gamma \\ &\text{subject to} && \max_{l \in \{1,2,\ldots,n_{\mathrm{seq}}\}} J^{(l)}_{\mathrm{q,rob}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \leq \gamma \\ & && \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{2.51}$$

In order to eliminate the *max* operator from the inequalities, the first inequality of the equivalent optimization problem

$$\begin{aligned} \underset{\boldsymbol{U}[k],\gamma}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & J_{\text{q,rob}}^{(l)}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\text{r}}[k]) \leq \gamma, \quad \forall l \in \{1, 2, \ldots, n_{\text{seq}}\}, \\ & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k], \end{aligned} \tag{2.52}$$

must hold *for all* $l \in \{1, 2, \ldots, n_{\text{seq}}\}$, which means that the number of inequalities increases. Nevertheless, the absence of the *max* operator allows bringing the optimization problem to a standard form. Using the specific cost function formulation (2.49), the optimization problem reads as

$$\begin{aligned} \underset{\boldsymbol{U}[k],\gamma}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & \left\| \boldsymbol{F}_{\text{q}}^{(l)} \boldsymbol{U}[k] + \boldsymbol{g}_{\text{q}}^{(l)}[k] \right\|_2^2 \leq \gamma, \quad \forall l \in \{1, 2, \ldots, n_{\text{seq}}\}, \\ & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k], \end{aligned} \tag{2.53}$$

which is transformed once more to get second-order cone constraints[4] in the optimization problem

$$\begin{aligned} \underset{\boldsymbol{U}[k],\gamma}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & \left\| \begin{bmatrix} 2\left(\boldsymbol{F}_{\text{q}}^{(l)} \boldsymbol{U}[k] + \boldsymbol{g}_{\text{q}}^{(l)}[k] \right) \\ 1 - \gamma \end{bmatrix} \right\|_2 \leq 1 + \gamma, \quad \forall l \in \{1, 2, \ldots, n_{\text{seq}}\}, \\ & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{2.54}$$

This optimization problem is the key ingredient of RMPC and now has the form of an SOCP, which represents a convex optimization problem. This problem is solved at every sampling instant k while the first entry of the resulting optimal input vector $\boldsymbol{U}^{\star}[k]$, namely $\boldsymbol{u}^{\star}[k]$, is then applied to the plant. The remaining transformation to the exact standard form (see (2.6)) is omitted here for reasons of brevity. The optimization problem (2.54) can be considered the robust counterpart of (2.32). It is emphasized that the weighting matrices $\boldsymbol{Q}_{\text{q}}$ and $\boldsymbol{R}_{\text{q}}$ remain the same as in the MPC formulation—no re-tuning is necessary. However, the RMPC formulation requires an SOCP solver in contrast to the MPC formulation, which requires a QP solver.

[4] This transformation is necessary because (2.53) contains a quadratic form in the constraints. In order to apply a standard SOCP solver this quadratic form needs to be expressed as a second-order cone.

2.3.3 Stability

Ensuring stability for RMPC, where uncertain parameters are involved, is even more complex than for MPC (Mayne et al., 2000). The basic principle is to add a terminal constraint and a terminal cost to the respective optimization problem as in MPC. Zheng (1995) introduced robust contraction constraints for stable plants. These constraints ensure that the states converge to an equilibrium *for all* possible uncertainty realizations. Another approach is the use of so-called robustly invariant terminal sets (Bemporad and Morari, 1999), which also guarantee robust stability. More details about robust stability are given in the works of Rawlings and Mayne (2009, p. 213ff.) and Bemporad and Morari (1999). However, the approaches to ensure robust stability are again mostly for *regulation* problems. In order to track arbitrary reference signals, once more a trend towards the decoupling of the stabilization of a possibly unstable plant and handling of constraints through reference governors (RGs) can be observed (Mayne et al., 2000).

2.4 Reference Governors

The general aim of reference governors (RGs) is to modify an existing arbitrary varying reference signal such that constraints of an underlying control system are respected. This implies that if the reference signal can be tracked by the underlying control system without violating constraints, the reference is *not* adapted by the RG (Kolmanovsky et al., 2012). In contrast to MPC, an RG is an add-on to a probably existing feedback controller—an MPC scheme replaces an existing controller.

Considering a reference in the shape of a ramp that is fed to an RG, the exact tracking of this exemplary reference signal would require a very high acceleration[5] over one sampling interval of a discrete-time control system. In practice, the acceleration is limited and cannot be changed instantly. An RG adapts the reference signal, i.e. “smoothens”, the ramp such that the underlying controller maintains the constraints and thus is able to track the adapted reference.

A stable design of MPC and RMPC schemes for unstable systems is quite complex (Mayne et al., 2000), as the topics of stabilization, control performance, and constraint satisfaction are handled by a *single* controller. This is the point where the concept of RGs leads to simplifications in the controller design phase. The underlying feedback controller is designed to achieve a stable closed-loop system and a good tracking behavior in the absence of constraints, whereas the RG ensures that system constraints are not violated.

[5] In the continuous-time case, even an infinite acceleration would be necessary.

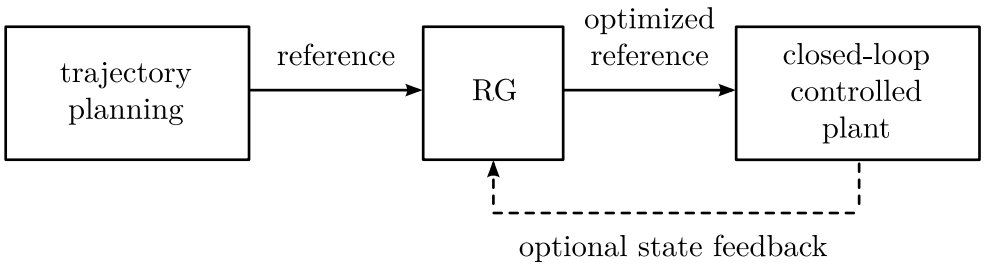


Figure 2.10: A basic reference governor (RG) structure

The basic structure of a system that includes an RG is depicted in Figure 2.10. The trajectory planning part delivers the reference signal that consists of simple shapes (e.g., steps or ramps) or more sophisticated shapes (e.g., acceleration- and jerk-limited trajectories).

The original concept of an RG presented in (Gilbert and Tan, 1995) is inspired by the maximal output admissible set theory developed by Gilbert and Tan (1991). The initial state of an LTI system without an input (unforced) is said to be output admissible with respect to an output constraint if the respective unforced output response does not violate the constraint. The set of all possible initial conditions that fulfill this definition is called maximal output admissible set. The first RG by Gilbert and Tan (1995) makes use of this theory to adapt the reference if constraints would be exceeded. Simultaneously, but independently, Bemporad and Mosca (1995) developed a predictive reference governor (PRG) that relies on some ideas of MPC. Nowadays, the concept of RGs often blends into MPC (Kolmanovsky et al., 2012).

Many variants of RGs have been proposed since their development. The RG variants differ in some key properties. For instance, Kogiso and Hirata (2006) presented an RG that can cope with an arbitrary reference, whereas Aghaei et al. (2013) treated constant references. A further objective that can be considered is robustness, which is treated in (Bemporad and Mosca, 1998; Guzman et al., 2009; Casavola et al., 2000b). Another point of distinction is whether the RG receives feedback of the plant as in (Gilbert and Tan, 1995; Bemporad and Mosca, 1995) or if no feedback is used (Sugie and Yamamoto, 2001; Hirata and Kogiso, 2001). Further recent developments in the field of RGs can be found in the work of Kolmanovsky et al. (2012).

3 The Concept of Predictive Reference Governors (PRGs)

This chapter introduces the concept of predictive reference governors (PRGs). Subsequent chapters treat several different variants of PRGs including near minimum-time and multiple-input, multiple-output (MIMO) formulations.

PRGs were introduced by Bemporad and Mosca (1995). A PRG is a combination of the concept of a reference governor (RG) presented in Section 2.4 and the ideas of model predictive control (MPC) (see Section 2.2). The main task of a PRG remains the modification of a reference signal in order to satisfy constraints that are present in the underlying control system. The theory of maximal output admissible sets is often used for the design of classical RGs, whereas PRGs rely on the concept of MPC to deal with constraints and simultaneously minimize the deviation of the reference signal by making use of a cost function. The PRG approach comes with the same advantages as a classical RG: The two important control objectives of constraint handling and stabilization of a possibly unstable plant are decoupled, which simplifies the overall design procedure. A PRG can be seen as an add-on to an existing (proven) feedback controller—an MPC scheme would completely replace the existing feedback controller. Traditionally, PRG formulations rely on quadratic form based cost functions: Aghaei et al. (2013) showed simulation results for constant reference signals and Stoican et al. (2012) used the PRG concept for fault detection purposes while tracking an arbitrary varying reference. Huber et al. (2013) used nonlinear MPC in a scheme they call model control loop, which can also be interpreted as a PRG approach.

Figure 3.1 illustrates the concept of the proposed PRG scheme. The structure consists of three main parts: The trajectory planning section, the PRG, and a two-degrees-of-freedom (2-DoF) based control scheme with a sampling time T_s. The 2-DoF based scheme, which is presented in detail in Chapter 4, consists of the dynamic, model-based feedforward control (DynFF) part and the feedback controlled plant. The part consisting of the PRG can be executed with a sampling time of $n_\mathrm{mr} \cdot T_\mathrm{s}$ (with $n_\mathrm{mr} \in \mathbb{N}$), which makes the overall structure a multi-rate control scheme, or even offline if the computational requirements are too high.

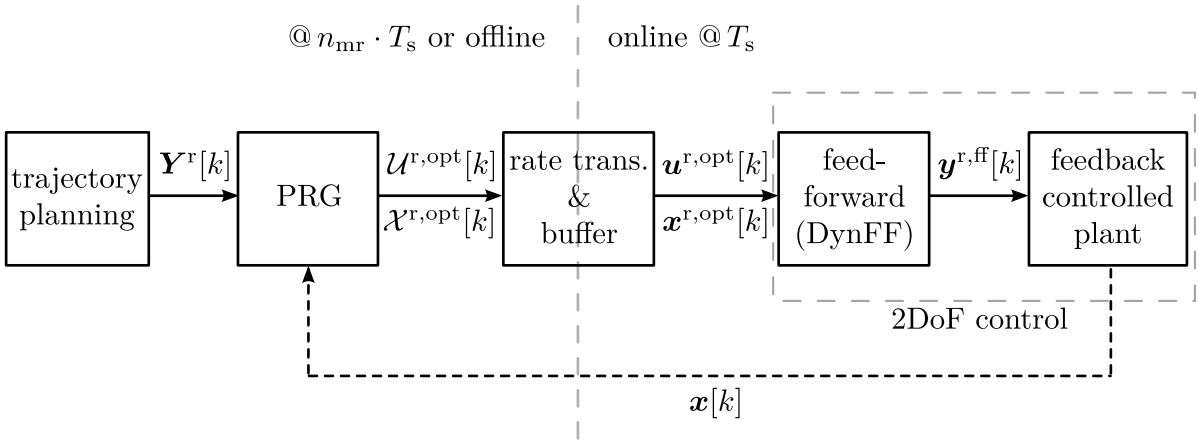


Figure 3.1: Detailed predictive reference governor (PRG) structure

The trajectory planning section generates the reference trajectory. This reference trajectory can be, for example, a step, a trajectory in the shape of a ramp, or a completely arbitrary varying reference signal. The stacked reference vector $\boldsymbol{Y}^{\mathrm{r}}[k]$ is delivered by the trajectory planning section to the PRG. As discussed in Section 2.2.2, $\boldsymbol{Y}^{\mathrm{r}}[k]$ is assumed to be constant if the reference is not known in advance and $\boldsymbol{Y}^{\mathrm{r}}[k]$ contains future reference values if they are known in advance (preview = true).

The PRG in Figure 3.1 is the key component of the whole structure. It is responsible for modifying the reference trajectory such that constraints in the closed-loop control system are maintained and that the deviation from the reference is as small as possible. If $\boldsymbol{Y}^{\mathrm{r}}$ can be tracked by the underlying controller without violating the constraints, $\boldsymbol{Y}^{\mathrm{r}}$ is *not* modified. The concept of MPC is perfectly suited for the needs of a PRG, as a cost function can be minimized in the presence of constraints. The general optimization problem of a PRG reads as

$$\begin{aligned} &\underset{\boldsymbol{U}[k]}{\text{minimize}} \quad J(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \\ &\text{subject to} \quad \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{3.1}$$

The optimization problem (3.1) is up to now quite general[1]—the only prerequisites are that it is convex and that it can be solved efficiently. Examples for the general optimization problem (3.1) are a quadratic program (QP) that arises in MPC (see (2.32)) or a second-order cone program (SOCP) that arises in RMPC (see (2.54))—the ongoing chapters deal with different types of cost functions and define the respective optimization problems more precisely.

The optimization problem (3.1) is solved in a receding horizon fashion as it is done in MPC. The optimal value of the optimization variable is denoted by $\boldsymbol{U}^{\star}[k]$.

[1]The element-wise constraints $\mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]$ serve as an example here.

The choice of the prediction horizon N depends on the settling time of the plant for the worst-case reference signal; N should be higher than the settling time to cover the complete dynamic behavior of the plant. A choice like this leads to coinciding open-loop and closed-loop trajectories (see Section 2.2.7) and therefore to optimal trajectories with respect to the cost function for the open- *and* closed-loop response.

Furthermore, a long prediction horizon is in close relationship with a stable overall PRG structure, because a long prediction horizon emulates infinite horizon control with its advantages of stability and robustness (Mayne et al., 2000). Well-known approaches from MPC like the use of terminal constraints and a terminal cost can be used to design a stable PRG—stability can be proofed for *regulation* problems (Mayne et al., 2000). A stability proof for a PRG that tracks of a constant reference is given in (Bemporad, 1997). As the PRGs in this work deal with the tracking of *arbitrary* varying reference signals, a rigorous proof of stability of the *overall* system cannot be given here. Nevertheless, compared with MPC, a PRG does not have to deal with the stabilization of a possibly unstable plant, as the plant is already stabilized by a feedback controller. The design of a stable overall system that includes the PRG and the stable closed-loop system is considered to be simplified compared with the design of a *single* MPC scheme.

The tuning of cost function weights is mainly done heuristically, as is the case for MPC. Keeping the number of tuning parameters as low as possible simplifies the tuning.

The overall PRG scheme presented in Figure 3.1 is a multi-rate scheme, which means that the PRG can operate with a higher sampling time $n_{\mathrm{mr}} \cdot T_{\mathrm{s}}$ than the digital control system that operates with the sampling time T_{s}. The factor $n_{\mathrm{mr}} \in \mathbb{N}$ is mainly dependent on the time it takes to solve the optimization problem (3.1) and therefore dependent on the problem type, the problem size and the used solver. If the sampling times in the overall scheme differ ($n_{\mathrm{mr}} > 1$), the optimization problem (3.1) is solved with a sampling time of $n_{\mathrm{mr}} \cdot T_{\mathrm{s}}$ and is therefore the receding horizon needs to be shifted by $N_{\mathrm{shift}} = n_{\mathrm{mr}} \in \mathbb{N}$. In the extreme case where n_{mr} is required to be larger than the prediction horizon N, the PRG scheme needs to be evaluated offline. This means that the whole reference trajectory is optimized offline; *afterwards*, the optimized trajectory is tracked by the controller. If the PRG is working offline, N_{shift} can be chosen in the range between 1 and the prediction horizon N. Increasing the value of the prediction horizon shift N_{shift} leads to a decreased computation time of the whole trajectory. Nevertheless, regarding the preview feature, a value of N_{shift} that is close to the prediction horizon might lead to suboptimal results, which is illustrated in Figure 3.2 by an example.

If the PRG is executed offline (especially for fast sampling systems), it is not possible to take the measured or estimated state $\boldsymbol{x}[k]$ into account. In this case, the state predic-

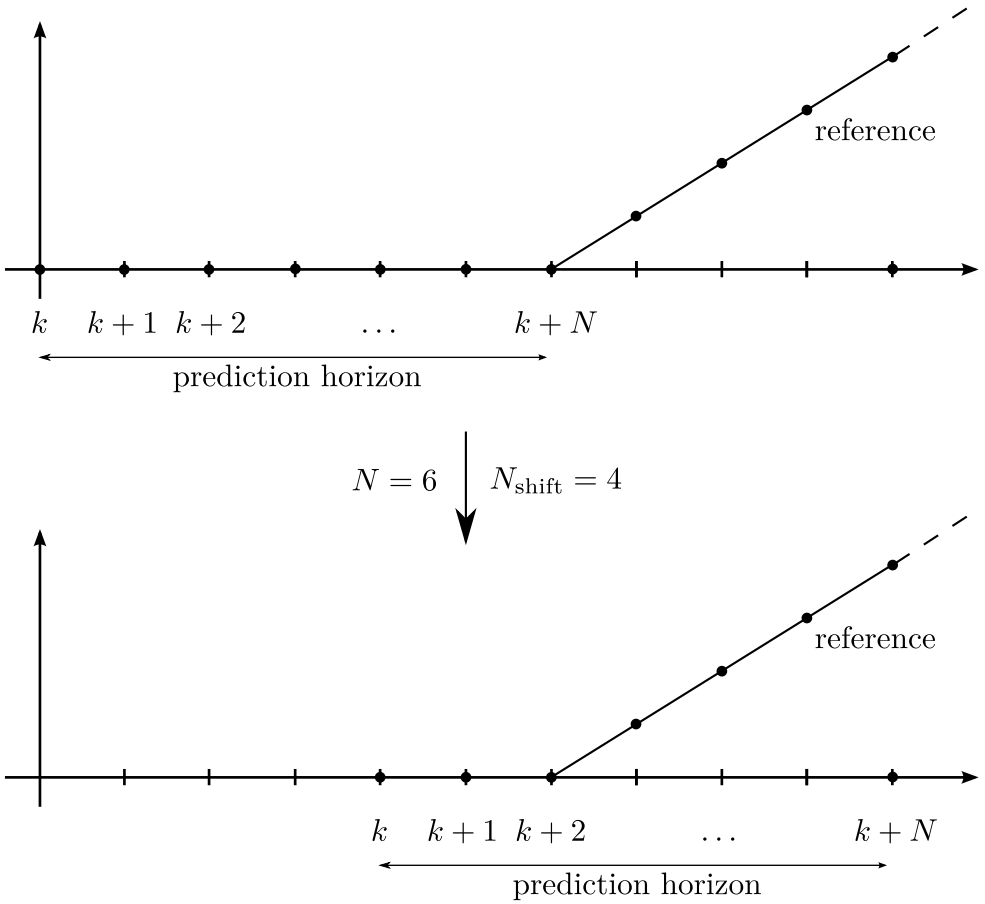


Figure 3.2: Choice of the prediction horizon shift N_{shift} for a PRG with preview. The top figure shows the initial iteration, where the reference in the shape of a ramp is just not "visible" for the PRG. The bottom figure shows the next iteration, after the prediction horizon $\mathbb{N} = 6$ has been shifted for $N_{\text{shift}} = 4$ sampling instants. Through this relatively large shift compared with the prediction horizon it might not be possible to react on the changing reference in time. This means the resulting output might significantly differ from the case $N_{\text{shift}} = 1$.

tion (2.15) is used to "simulate" the plant to determine the state $\boldsymbol{x}_{\text{pred}} = \boldsymbol{x}[k+N_{\text{shift}}]$ which is then used as the actual system state in the next iteration. In the presence of an external disturbance, model mismatch, or unmodeled phenomena the feedback controller is responsible for steering back the output to its reference trajectory.

The term *receding horizon* is also used in this thesis if it is larger than one, which is the standard case in MPC, and also if no feedback is available.

The overall principle of the PRG scheme is also shown as pseudo-code in Algorithm 1 (see Appendix A). The *main loop* that is executed with a sampling time of $n_{\text{mr}} \cdot T_{\text{s}}$ or offline (without a predefined sampling rate) is shown in Algorithm 2 (see Appendix A).

The trajectories of the optimized inputs and states, $\mathcal{U}^{\text{r,opt}}[k]$ and $\mathcal{X}^{\text{r,opt}}[k]$, are resulting from solving the optimization problem (3.1) and are gathered in a receding horizon fashion. As the system output is a linear combination of the system states, the output trajectory

is included in $\mathcal{X}^{\mathrm{r,opt}}[k]$. In this work, however, the output trajectory is not used directly. Chapter 4 shows that the inherently available signals $\mathcal{U}^{\mathrm{r,opt}}[k]$ and $\mathcal{X}^{\mathrm{r,opt}}[k]$ can be efficiently used as feed-forward signals in a 2-DoF controller using the so-called DynFF concept (Roppenecker, 2009). Figure 3.1 already includes this feedforward strategy between the PRG and the closed-loop controlled plant.

The PRG algorithm that is presented here uses the open-loop model (2.12) without a feedback controller for the formulation of the optimization problem—the reason for this is the close relationship to MPC. However, if the open-loop system is unstable, it is necessary to use a stable closed-loop model that includes the feedback controller for the PRG design. This can be achieved through slight modifications that are sketched in Section 4.2.2.1, where the feedback controller design is discussed.

The proposed multi-rate PRG scheme allows to scale the sampling times in order to evaluate the optimization algorithm in real-time, or, in the extreme case, even offline. This flexibility enables the use of a long prediction horizon, which ensures closed-loop optimality and contributes to the overall stability of the control scheme. The offline application of a PRG allows to exploit the features of MPC, like the handling of constraints and the minimization of a cost function, even for fast sampling systems that do not yet allow operating a PRG scheme in real-time.

4 Modeling and Control Structure of an Industrial Positioning System

The common demand in many processes for an increased speed/productivity and an increased precision is the driving force behind model-based control methods. Simple structures, for example consisting of proportional-integral-derivative (PID) or linear state-space controllers, can often not fulfill these modern control requirements, especially in the presence of constraints.

The predictive reference governor (PRG) schemes presented in this work are validated with a highly dynamic industrial laser positioning system. Such positioning systems are needed, for example, in the fields of laser welding and cutting, rapid prototyping, or in medical technology. A schematic drawing of this system is presented in Figure 4.1. Deflection of a laser in two dimensions is achieved with mirrors that are mounted on two independent electric motors. This setup allows positioning a laser beam with high speed, as the involved motor-mirror combinations offer low moments of inertia compared to the positioning of the whole laser unit. The respective electric motor is a single-phase motor with a permanent magnet (PM) on its rotor. As the motor in such a positioning system is only required to rotate about $\pm 10^\circ$ there is no need for a commutator. A position detector placed at the end of the motor (opposite to the mirror) is used to measure the angular position of the rotor. This position detector together with a current sensor and a digital control system form a servomechanism. The high dynamic and precision requirements of these applications demand fast sampling control systems and precise modeling of the electromechanical system. Due to finite stiffness, the mechanical system of the mirror and the rotor is modeled as a two-mass system that is connected through a spring-damper system. This means that the angular load position[1] (mirror) and the angular motor position differ especially for fast positioning. As the sensor (position detector) and the actuator (motor) are placed in the same location—assuming a stiff coupling between the position detector and the rotor—this control scheme is called collocated control. If the angular load position would be measured, the sensor and the actuator would be in different locations, which would lead to non-collocated

[1] Angular *load* position is used from here on synonymously with angular *mirror* position, in order to stick to the common "motor/load" notation.

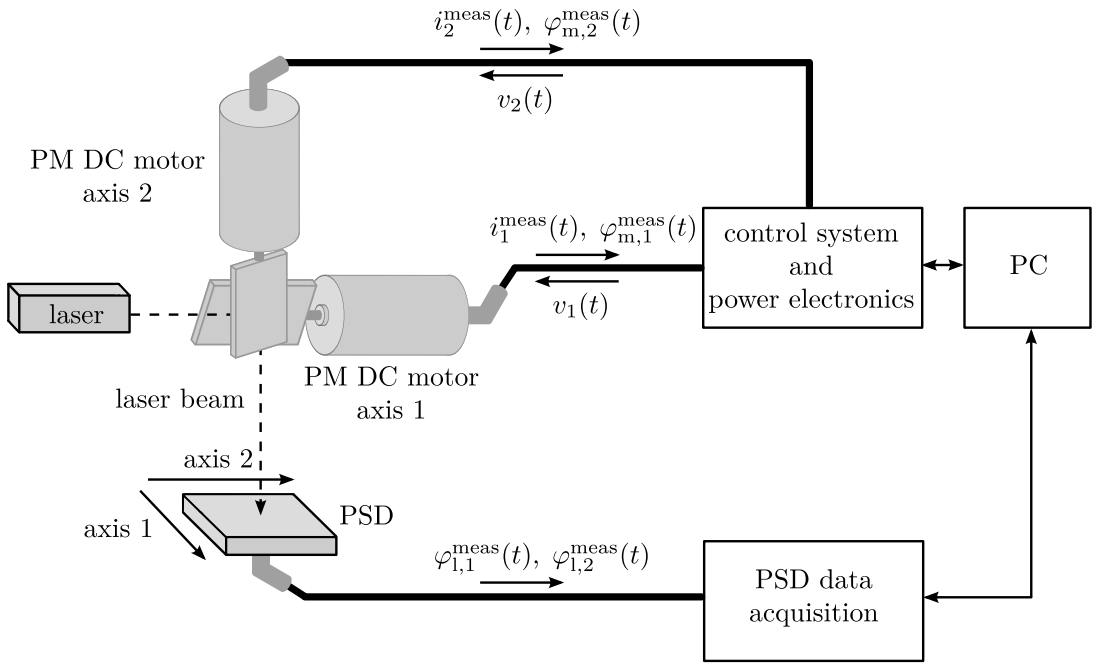


Figure 4.1: Positioning system with the optical path, position sensitive device (PSD), and control system. The control system measures the angular motor positions ($\varphi_{\text{m},1}^{\text{meas}}$ and $\varphi_{\text{m},2}^{\text{meas}}$) and the currents ($i_1^{\text{meas}}$ and i_2^{meas}). The input voltages of the motors are denoted by v_1 and v_2, respectively. The position sensitive device (PSD) measures the spot position, which is converted to the measured angular load positions $\varphi_{\text{l},1}^{\text{meas}}$ and $\varphi_{\text{l},2}^{\text{meas}}$. This experimental setup is only used for validation purposes—in normal operation, the PSD is replaced by a workpiece.

control (Preumont, 2011, p. 117ff.). The performance measure of a laser positioning system is the position of the laser spot, which is typically focused on a workpiece. Optical measurement systems, for example a camera, are necessary to measure the position of the spot. However, these measurement systems often have a slow dynamic response, which make them unusable for control purposes. The spot position is dependent on the two angular load positions (mirrors) by a nonlinear geometric transformation (Tang et al., 2004). This position transformation in turn is given by the mechanical design of the positioning system and is used to determine the angular load positions that correspond to a given spot position. Hence, it is possible to control the spot position by controlling the respective load positions. Nevertheless, it is also hard to *measure* the load positions without a high additional cost and effort. In order to bypass the problems of measuring the spot or the load positions, in this work the load position is estimated in order to use this position as the output that is controlled.

The following sections describe how the electromechanical system is modeled, identified, and controlled. They also show that it is reasonable to use the estimated load position for control. Measurements with a PSD, which allows measuring the laser spot position, validate this approach.

4.1 System Modeling and Identification

The foundation of model-based control (e.g., state-space control, model predictive control (MPC), etc.) is a system model that is detailed enough to represent the most important system dynamics.

The considered application consists of two permanent magnet (PM) direct current (DC) motors with mirrors attached to them. The following remarks concerning the modeling and the identification of the system parameters deal with a single motor. The principles, however, apply to both used motors, as they just differ in the system parameters and not in their general design.

The PM DC motor is modeled according to standard references (Schröder, 2013; Isermann, 2005). The stator winding of the single-phase motor is modeled by an inductance L in series with a resistance R. The current in the stator windings $i(t)$ interacts with the part of the magnetic field of the PM on the rotor that is perpendicular to the current and thus generates a radial force, which leads to a torque; this interaction is modeled through the torque constant K_t. The fact that this motor has no commutator and only one pole pair makes the torque "constant" dependent on the position $K_\mathrm{t}(\varphi_\mathrm{m}(t)) = \Psi_\mathrm{pm}\cos(\varphi_\mathrm{m}(t))$, where Ψ_pm is the flux linkage of the permanent magnet. This position dependency is neglected ($K_\mathrm{t} = \Psi_\mathrm{pm} = \text{constant}$) in this work because of the relatively small angles of $\pm 10°$ that are needed for the specific positioning application—the maximum relative error of the torque constant K_t in this position range is $\pm 1.52\,\%$. The generated torque can accelerate the rotor and thus influences the motor position $\varphi_\mathrm{m}(t)$ and motor speed $\omega_\mathrm{m}(t)$. The resulting induced voltage in the stator winding (back electromotive force (EMF)) is modeled as $K_\mathrm{t}\omega_\mathrm{m}(t)$. This back EMF counteracts the input voltage $v(t)$. The unit of the torque constant K_t is $\mathrm{N\,m\,A^{-1}}$, which corresponds to $\mathrm{V\,s}$.

The input voltage $v(t)$ is applied to the windings through pulse width modulation (PWM) together with an H-bridge consisting of four MOS-FETs. The influence of this power electronics based actuator (dead-times, switching times, etc.) is neglected in the modeling of the system, as the involved PWM sampling time $T_\mathrm{PWM} = 1/f_\mathrm{PWM} = 3.33\,\mu\mathrm{s}$ is much lower than the electrical time constant $\tau_\mathrm{el} = L/R \approx 60\,\mu\mathrm{s}$.

Supping up, the differential equation of the current $i(t)$ (electrical subsystem) reads as

$$L\frac{\mathrm{d}}{\mathrm{d}t}i(t) = -Ri(t) - K_\mathrm{t}\omega_\mathrm{m}(t) + v(t). \tag{4.1}$$

The acceleration is dependent on the torque and the moment of inertia. The topic of friction plays a minor role for the application that is considered here; the involved moment of inertia is small compared to the available torque. Hence, friction is modeled as simple

viscous friction depending on the motor speed $\omega_\mathrm{m}(t)$ and the viscous friction coefficient K_f. More advanced friction models can be found, for example, in (Olsson et al., 1998) or (Hackl, 2012, p. 17–28).

The final part in the modeling of the plant is the interaction between the load and the motor. The load position and the load speed are denoted by $\varphi_\mathrm{l}(t)$ and $\omega_\mathrm{l}(t)$, respectively. The high dynamic capabilities of the motor together with the finite stiffness of the rotor-load system make it necessary to model this mechanical system as a two-mass system consisting of the motor moment of inertia J_m and the load moment of inertia J_l (Ellis, 2000, p. 331–349). The connection of the motor and the load is modeled by a spring-damper system with the spring constant c and the damping d.

The differential equations of the mechanical subsystem are

$$\begin{aligned} J_\mathrm{m}\frac{\mathrm{d}}{\mathrm{d}t}\omega_\mathrm{m}(t) &= K_\mathrm{t} i(t) - c\varphi_\mathrm{m}(t) - (d + K_\mathrm{f})\,\omega_\mathrm{m}(t) + c\varphi_\mathrm{l}(t) + d\omega_\mathrm{l}(t), \\ J_\mathrm{l}\frac{\mathrm{d}}{\mathrm{d}t}\omega_\mathrm{l}(t) &= c\varphi_\mathrm{m}(t) + d\omega_\mathrm{m}(t) - c\varphi_\mathrm{l}(t) - d\omega_\mathrm{l}(t), \end{aligned} \tag{4.2}$$

with

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\varphi_\mathrm{m}(t) &= \omega_\mathrm{m}(t), \\ \frac{\mathrm{d}}{\mathrm{d}t}\varphi_\mathrm{l}(t) &= \omega_\mathrm{l}(t). \end{aligned} \tag{4.3}$$

The differential equations that govern the discussed subsystems of the overall model are brought together to a general continuous-time multiple-input, multiple-output (MIMO) linear time-invariant (LTI) system in the form of

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}_c\boldsymbol{x}(t) + \boldsymbol{B}_c\boldsymbol{u}(t), \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t), \end{aligned} \tag{4.4}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ is the system input vector, and $\boldsymbol{y}(t) \in \mathbb{R}^{n_y}$ is the output vector. The numbers of system states, inputs, and outputs are n_x, n_u and n_y, respectively. The matrices $\boldsymbol{A}_c \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{B}_c \in \mathbb{R}^{n_x \times n_u}$, and $\boldsymbol{C} \in \mathbb{R}^{n_y \times n_x}$ are the continuous-time state matrix, the input matrix, and the output matrix, respectively.

The state vector $\boldsymbol{x}(t)$ and the input vector $\boldsymbol{u}(t)$ for the PM DC motor are written as

$$\begin{aligned} \boldsymbol{x}(t) &= \begin{bmatrix} i(t) & \varphi_\mathrm{m}(t) & \omega_\mathrm{m}(t) & \varphi_\mathrm{l}(t) & \omega_\mathrm{l}(t) \end{bmatrix}^\top, \\ \boldsymbol{u}(t) &= \begin{bmatrix} v(t) \end{bmatrix}. \end{aligned} \tag{4.5}$$

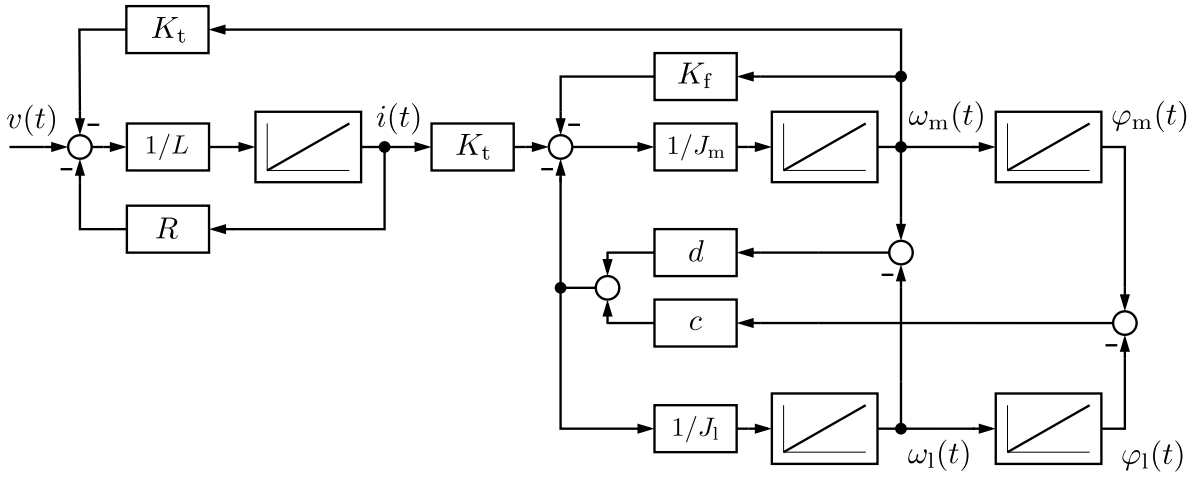


Figure 4.2: Signal flow graph of the continuous-time LTI model of a single PM DC motor that forms a two-mass system with the load

The output vector $\boldsymbol{y}(t)$, which consists of the measurable states, is defined as

$$\boldsymbol{y}(t) = \begin{bmatrix} i(t) & \varphi_\mathrm{m}(t) \end{bmatrix}^\top . \tag{4.6}$$

The dimensions of the input ($n_u = 1$) and output vector ($n_y = 2$) suggest that the system considered here is a single-input, multiple-output (SIMO) system which is a subclass of the more general MIMO systems.

Considering (4.1)–(4.3), the matrices of the state-space representation are

$$\boldsymbol{A}_c = \begin{bmatrix} -\frac{R}{L} & 0 & -\frac{K_\mathrm{t}}{L} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \frac{K_\mathrm{t}}{J_\mathrm{m}} & -\frac{c}{J_\mathrm{m}} & -\frac{d+K_\mathrm{f}}{J_\mathrm{m}} & \frac{c}{J_\mathrm{m}} & \frac{d}{J_\mathrm{m}} \\ 0 & 0 & 0 & 0 & 1 \\ 0 & \frac{c}{J_\mathrm{l}} & \frac{d}{J_\mathrm{l}} & -\frac{c}{J_\mathrm{l}} & -\frac{d}{J_\mathrm{l}} \end{bmatrix}, \quad \boldsymbol{B}_c = \begin{bmatrix} \frac{1}{L} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} . \tag{4.7}$$

The system output matrix that follows from (4.5) and (4.6) is

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} . \tag{4.8}$$

The output defined by (4.6) is needed for system identification and state estimation. However, for feedback control and for the predictive reference governor (PRG) approaches another choice of the output $\boldsymbol{y}(t)$ is necessary in order to control the load position $\varphi_\mathrm{l}(t)$. The system flow graph of the system described by (4.4)–(4.8) is shown in Figure 4.2.

Figure 4.3 schematically illustrates the most important parts of the system that is controlled. It includes—besides the PM DC motor and the load—the PWM, the power electron-

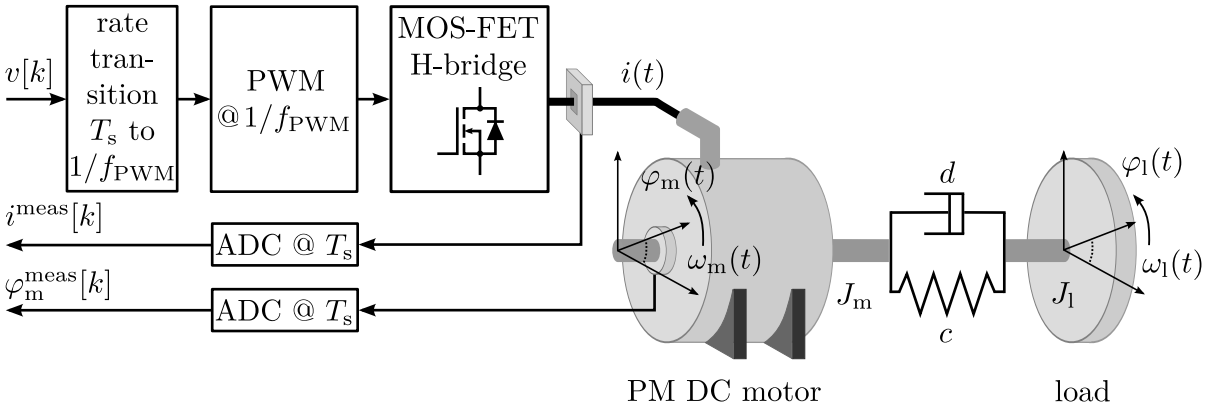


Figure 4.3: Schematic representation of a single PM DC motor that forms a two-mass system with the load. This structure represents the discrete-time plant that is "seen" by the digital control system.

ics, and the data acquisition (measurement of $i^{\text{meas}}[k]$, $\varphi_{\text{m}}^{\text{meas}}[k]$). Hence, this figure represents the plant that is "seen" by the digital control system.

The discrete-time nature of the digital control system makes it necessary to transform the matrices $\boldsymbol{A}_c$ and $\boldsymbol{B}_c$ to their discrete equivalents $\boldsymbol{A}$ and $\boldsymbol{B}$. The general discrete-time MIMO LTI model (2.12) is repeated here for reasons of completeness

$$\begin{aligned} \boldsymbol{x}[k+1] &= \boldsymbol{A}\boldsymbol{x}[k] + \boldsymbol{B}\boldsymbol{u}[k], \\ \boldsymbol{y}[k] &= \boldsymbol{C}\boldsymbol{x}[k]. \end{aligned} \tag{4.9}$$

The discrete equivalents $\boldsymbol{A}$ and $\boldsymbol{B}$ are determined using the zero-order hold (ZOH) discretization (Franklin et al., 1997, p. 203) to account for a constant plant input $\boldsymbol{u}[k]$ between two sampling instants. The discrete matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are determined by the ZOH discretization such that the states of the continuous-time model (4.4) and the states of the discrete-time model (4.9) are equal at the sampling instants for a constant input $\boldsymbol{u}[k]$. Furthermore, the ZOH discretization ensures stability of the discrete-time system if the continuous-time system is stable. Other discretization methods like the popular and simple Euler forward approximation can lead to an unstable discrete-time system even if the continuous-time system is stable. Nevertheless, the Euler forward approximation can lead to a discrete-time model that is in good agreement with the continuous-time model if the controller sampling time T_s is small compared with the smallest system time constant. In this work, however, the ZOH discretization is used because the smallest system time constant, the electrical time constant $\tau_{\text{el}} = L/R \approx 60\,\mu\text{s}$, is already in the range of the controller sampling time $T_s = 10\,\mu\text{s}$, which cannot be lowered further due to computational constraints. The output $\boldsymbol{y}[k]$ and the output matrix $\boldsymbol{C}$ remain the same as for the continuous time-case (see (4.6) and (4.8)).

The discrete-time transfer function (TF) matrix $\boldsymbol{H}(z)$ of a system in the form of (4.9) reads as

$$\boldsymbol{H}(z) = \frac{\boldsymbol{X}^{\mathrm{tf}}(z)}{\boldsymbol{U}^{\mathrm{tf}}(z)} = (z\boldsymbol{I}_{n_x \times n_x} - \boldsymbol{A})^{-1}\,\boldsymbol{B}, \tag{4.10}$$

where $\boldsymbol{U}^{\mathrm{tf}}(z)$ and $\boldsymbol{X}^{\mathrm{tf}}(z)$ are z-transforms of the input $\boldsymbol{u}[k]$ and the state $\boldsymbol{x}[k]$, respectively. Hence, the TF matrix $\boldsymbol{H}(z)$ contains *all* possible combinations of TFs from the inputs to the states. The TF from $v[k]$ to $\varphi_{\mathrm{l}}[k]$, for example, is denoted by $H_{v\to\varphi_{\mathrm{l}}}(z)$.

The experimental setup (see Figure 4.1) comprises a personal computer (PC) running Mathwork's MATLAB®. The computer interacts with a digital signal processor (DSP) through a dynamic link library (DLL). This DSP is operated in real-time and is responsible for control, measurement, and communication. Hence, the experimental results (for identification, analysis, etc.) are directly available in MATLAB.

The next step is to identify the parameters of the continuous-time state-space matrices $\boldsymbol{A}_c$ and $\boldsymbol{B}_c$, which subsequently lead to their discrete-time equivalents $\boldsymbol{A}$ and $\boldsymbol{B}$. An important difference of the application considered here—in comparison to most standard control systems—is that the state that needs to be controlled, the load position $\varphi_{\mathrm{l}}[k]$, is *not* measured. In the case where this state can be measured, it is enough to identify a system model that reproduces the input-output behavior of the system. This means the model does not reproduce physically motivated internal states. This reproducibility, however, is essential for this work because it is desired to control the load position $\varphi_{\mathrm{l}}[k]$ while only the measurements of the current $i^{\mathrm{meas}}[k]$ and the motor position $\varphi_{\mathrm{m}}^{\mathrm{meas}}[k]$ are available. Furthermore, if states that are not measured are required to respect constraints, it is also essential to keep the physical connection between the states in the identification procedure. Summing up, it is necessary to identify the physical parameters of the state-space matrices $\boldsymbol{A}_c$ and $\boldsymbol{B}_c$ in order to ensure the physical connection between the system states.

The two most general system identification approaches are called black box identification and grey box identification (Ljung, 1999; Isermann and Münchhof, 2011). Black box identification refers to the case where little or even no information about the system under control is available or to the case where the system is too complex to be modeled from a physical perspective. This would mean that *all* entries of $\boldsymbol{A}_{\mathrm{c}}$ and $\boldsymbol{B}_{\mathrm{c}}$ are identified without knowledge about the (physical) connection between these values; this in turn leads to (internal) states that do, in general, *not* correspond to physical values[2]. As already discussed, the physical connection between the system states is essential in this work. Hence, grey box identifica-

[2] A specific input-output behavior can be achieved through *infinitely* many state-space representations.

tion is used here, where a model is constructed that often relies on physical reasoning and contains free parameters that need to be identified.

The basic process of grey box system identification consists of the following steps:

- Define a model (often physics based) with free parameters.
- Collect training-data, a sequence of output values $\boldsymbol{y}^{\text{meas}}[k]$, using a suitable sequence of input values $\boldsymbol{u}[k]$ that excites all important system dynamics. In this case, the measured output is $\boldsymbol{y}^{\text{meas}}[k] = \left[\, i^{\text{meas}}[k] \;\; \varphi_{\text{m}}^{\text{meas}}[k] \,\right]^{\top}$ and the system input is $\boldsymbol{u}[k] = \left[\, v[k] \,\right]$.
- Estimate the free model parameters based on the sequences of the output error $\boldsymbol{y}^{\text{meas}}[k] - \boldsymbol{y}[k]$ and the input $\boldsymbol{u}[k]$, where $\boldsymbol{y}[k]$ is the output of the identified model. This parameter identification is often based on an unconstrained optimization problem.
- Validate the identified model using data that are *different* from the training-data and again excite all important system dynamics.

The identification of a rotational two-mass system is a quite common problem in literature. In the field of electrical drive systems, it is often essential to take into account the oscillatory behavior of two-mass systems. Endisch et al. (2009) used a dynamic neural network to identify two-mass parameters and a nonlinear friction model. An identification method in the frequency domain is presented in (Villwock and Pacas, 2008; Pacas et al., 2010). The so-called Welch method, which originates from the field of signal processing, is used to identify the frequency response of the system. The model that has a previously defined structure, but consists of the free parameters, is fitted to the identified frequency response. This process of model fitting is a nonlinear, unconstrained optimization problem, which is solved by using the Levenberg-Marquardt algorithm. An identification procedure based on the recursive least squares (RLS) algorithm was used by Saarakkala and Hinkkanen (2013); the collected discrete-time signals are processed by the RLS approach to gain a discrete-time TF. The gathered TF is then transformed to a continuous-time function and compared with the TF that consists of the free parameters. This comparison might lead to a non-unique result for the parameters; as shown in (Endisch, 2009, p. 225) the parameters of the mechanical subsystem of a two-mass system *cannot* be determined uniquely from a previously identified transfer function. An approach presented by Unterholzner and Wünsche (2009) relies on the RLS algorithm to identify the system parameters of a one-mass system *online* in an adaptive control scheme.

System identification is an important part of this work, as model-based control as well as constraint satisfaction require precise system models. The identification process, however, is not the focus of this thesis. Therefore, only the rough process of system identification that

is used here is described in the following. The system is operated in closed-loop in order to prevent damage, as the angular position is mechanically constrained. The first part of this identification method is the same as in (Saarakkala and Hinkkanen, 2013). Discrete-time sequences for $v[k]$, $i^{\mathrm{meas}}[k]$, and $\varphi_{\mathrm{m}}^{\mathrm{meas}}[k]$ are collected. The gathered sequences are used together with an RLS algorithm to determine two discrete-time TFs, namely, $H_{v\to i^{\mathrm{meas}}}(z)$ and $H_{v\to\varphi_{\mathrm{m}}^{\mathrm{meas}}}(z)$. Particle swarm optimization (PSO) (Kennedy and Eberhart, 1995) is then used in this work to fit the discrete-time TFs of the model ($H_{v\to i}(z)$ and $H_{v\to\varphi_{\mathrm{m}}}(z)$) to the two TFs gained through the RLS algorithm ($H_{v\to i^{\mathrm{meas}}}(z)$ and $H_{v\to\varphi_{\mathrm{m}}^{\mathrm{meas}}}(z)$). PSO is a promising approach for global and nonlinear optimization. It has the advantage that constraints on the parameters can be easily incorporated—for example, if values for nominal parameters are available, the parameters that are identified can be constrained to lie in a certain range around these nominal values. By using this approach, all free parameters of the two motors were identified, except the torque constant K_{t}, which was fixed (gained from a finite element analysis (FEA) of the motor) to prevent problems of parameter uniqueness (Endisch, 2009, p. 225). Furthermore, the viscous friction coefficient K_{f} was determined empirically[3].

The plant parameters, important control system parameters, and constraints are listed in Table 4.1. To distinguish between the system matrices of the two motors, a subscript identifies the respective axis (e.g., $\boldsymbol{A}_1$, $\boldsymbol{A}_2$, $\varphi_{\mathrm{l},1}[k]$, $\varphi_{\mathrm{l},2}[k]$). The maximum voltage $v_{\max}$ that is considered in the PRG and trajectory planning approaches is slightly below the supply voltage v_{DC} (hard constraint) to establish a control reserve (CR) $v_{\mathrm{DC}} - v_{\max}$ that is reserved for the feedback controller to react to disturbances or model mismatch. The maximum motor current $i_{\max}$ is based on thermal considerations and is seen as a soft constraint that is allowed to be exceeded for short periods of time. The reason for the rather different two-mass system parameters of both axes is the mirror design of axis 2, which is different to the design of axis 1 because a larger deflection area is needed on axis 2.

Considering the identified system parameters in Table 4.1, the respective state matrices ($\boldsymbol{A}_1$ and $\boldsymbol{A}_2$) have five eigenvalues. Figure 4.4 shows the pole zero map exemplary for the motor/axis 1. Nevertheless, the conclusions that can be drawn are the same for both motors. One eigenvalue lies exactly at 1 on the real axis, which makes the open-loop system *marginally stable*, as only one eigenvalue is on the unit-circle and the other eigenvalues are inside the unit-circle (Nise, 2011, p. 743). Another eigenvalue is located on the real axis very close to 1 inside the unit-circle[4]. The two eigenvalues that are at 1 or at least close to it represent the basic double integrator behavior of a positioning system. The reason why the

[3]The input signal that led to good PSO identification results led to small angular velocities. For the identification of the viscous friction coefficient K_{f}, however, comparably high velocities are necessary; this is the reason why K_{f} was determined empirically.

[4]The eigenvalue that is very close to 1 is not distinguishable from the eigenvalue that is exactly at 1 in Figure 4.4.

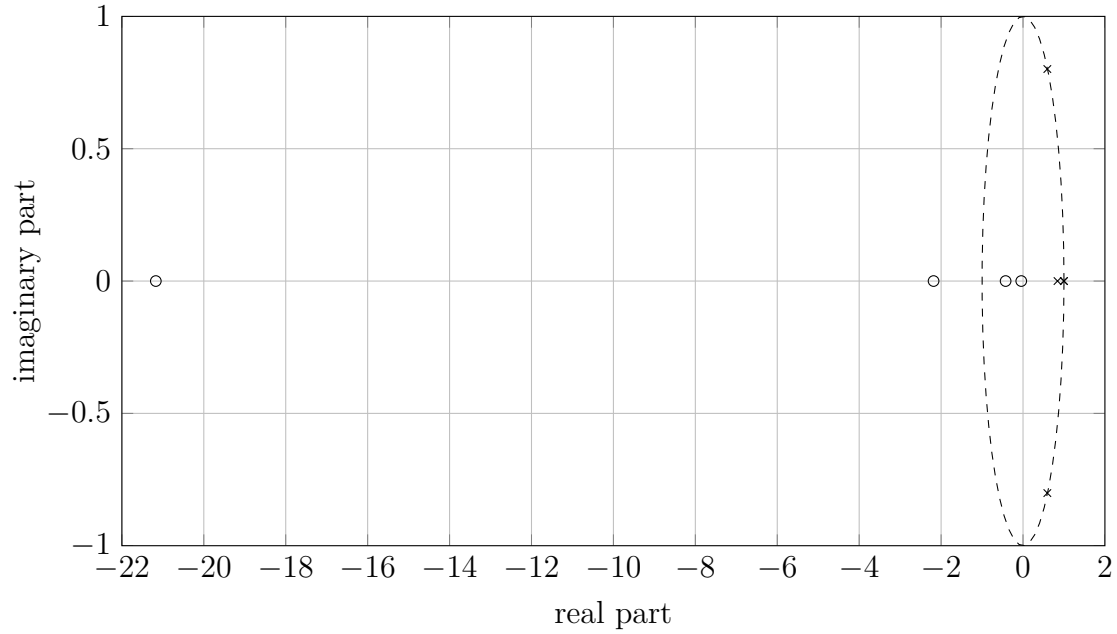


Figure 4.4: Pole zero map of the TF $H_{v_1 \to \varphi_{l,1}}(z)$ of the motor/axis 1: Poles (×), zeros (○), and unit-circle (- - - ·).

second eigenvalue is not exactly at 1 is the viscous friction and the back EMF. A system consisting of a pure double integrator, with two eigenvalues on the unit-circle, would be *unstable*. The third eigenvalue is located on the real axis inside the unit-circle and is caused by the dynamics of the electrical subsystem. The remaining two eigenvalues are complex and are caused by the non-ideal coupling of the load to the motor (two-mass system). These two complex eigenvalues lead, due to their proximity to the unit-circle, to slightly damped oscillations in the step response of the open-loop system. The discussed eigenvalues are the poles of the discrete-time TF $H_{v \to \varphi_l}(z)$, which represents the input-output behavior from the input voltage $v[k]$ to load position $\varphi_l[k]$. Another crucial property of TFs is the location of the zeros. For the identified parameters (see Table 4.1), the discrete-time TF $H_{v \to \varphi_l}(z)$ comprises four zeros (see Figure 4.4). The respective *continuous-time* TF, however, has only one zero. Hence, three of the four discrete-time zeros are introduced through the process of discretization of the continuous-time system model. These additional zeros are often called sampling zeros (Aström et al., 1984; Tesfaye and Tomizuka, 1995). Two of the sampling zeros lie *outside* the unit-circle of the complex plane and are therefore non-minimum-phase zeros (see Figure 4.4). Depending on the sampling time T_s, sampling zeros might even occur if the continuous time TF is minimum-phase. Non-minimum-phase zeros need special attention if a system inversion technique is used for feedforward control in a two-degrees-of-freedom (2-DoF) control structure.

Controller sampling time T_s	$10\,\mu\mathrm{s}$
PWM frequency f_PWM	$300\,\mathrm{kHz}$
Supply voltage v_DC	$29.4\,\mathrm{V}$
Maximum voltage v_max	$28\,\mathrm{V}$
Maximum current i_max	$5\,\mathrm{A}$

	Motor/axis 1	Motor/axis 2
Motor resistance R	$3.33\,\Omega$	$3.29\,\Omega$
Motor inductance L	$190.13\,\mu\mathrm{H}$	$190.43\,\mu\mathrm{H}$
Motor torque constant K_t	$6.40 \cdot 10^{-3}\,\mathrm{N\,m\,A^{-1}}$	$6.40 \cdot 10^{-3}\,\mathrm{N\,m\,A^{-1}}$
Motor moment of inertia J_m	$20.36 \cdot 10^{-9}\,\mathrm{kg\,m^2}$	$14.46 \cdot 10^{-9}\,\mathrm{kg\,m^2}$
Load moment of inertia J_l	$40.25 \cdot 10^{-9}\,\mathrm{kg\,m^2}$	$34.07 \cdot 10^{-9}\,\mathrm{kg\,m^2}$
Coupling spring constant c	$117.58\,\mathrm{N\,m\,rad^{-1}}$	$153.22\,\mathrm{N\,m\,rad^{-1}}$
Coupling damping constant d	$5.61 \cdot 10^{-6}\,\mathrm{N\,m\,s\,rad^{-1}}$	$9.35 \cdot 10^{-6}\,\mathrm{N\,m\,s\,rad^{-1}}$
Viscous friction coefficient K_f	$4 \cdot 10^{-6}\,\mathrm{N\,m\,s\,rad^{-1}}$	$4 \cdot 10^{-6}\,\mathrm{N\,m\,s\,rad^{-1}}$
Electrical time constant $\tau_\mathrm{el} = L/R$	$57.58\,\mu\mathrm{s}$	$57.88\,\mu\mathrm{s}$

Table 4.1: Control system and plant parameters for both PM DC motors

4.2 Control Structure

The overall control goal is to achieve a stable and fast closed-loop position control while respecting state and input constraints. Here, these constraints are voltage and current limitations that have to be maintained.

Classical control approaches for position control consist of an inner current/torque control loop, a speed controller, and an outer position control loop. These structures consist of cascaded proportional-integral-derivative (PID) controllers (or variants thereof, for example PI, PD, etc.) with limited outputs to account for constraints. The problem of integrator wind-up often arises in these approaches. Even schemes that are equipped with anti wind-up mechanisms might lead to unsatisfactory control results. Furthermore, when dealing with resonant systems, for example, notch-filters have to be introduced in order to prevent excitation of the resonance frequency. Another common but conservative approach is to keep the controller bandwidth low in order to avoid exciting the resonance of the two-mass system.

More advanced control methods for position control include, for example, the use of high-gain adaptive control for speed control of a two-mass system (Hackl, 2012, p. 199). This approach efficiently damps oscillations that are excited by the high-gain in a short amount of time, but it is hardly possible to consider constraints in the formulation of the control law. Another possibility to control a two-mass system is the use of a so-called biquad filter in the

feedback path (Alders et al., 2005). Due to its ability to handle constraints, MPC gained importance in the control of electrical drives. Modern MPC methods for electrical drives are roughly divided into two groups, namely standard MPC and direct MPC approaches. Standard MPC approaches (Linder et al., 2010; Bolognani et al., 2011) rely on the formulation presented in Section 2.2 and are based on *continuous valued* variables—a modulator is needed to drive the power electronics. Examples of standard MPC applied to two-mass systems can be found in (Serkies and Szabat, 2013; Szabat et al., 2010); both approaches were realized with explicit MPC (Herceg et al., 2013). Direct MPC methods (Karamanakos et al., 2014b; Rodriguez et al., 2013) are based on mixed integer optimization problems, which directly consider the switching states of the power electronics. These optimization problems are often solved by evaluating *all* possible switching states of the power electronics. A problem of direct methods—in contrast to standard MPC—is the exponentially growing computational effort with an increasing prediction horizon (if all switching states are evaluated). Karamanakos et al. (2014a), for example, decreased the computational burden by so-called sphere decoding—this approach is a kind of a branch and bound algorithm, which does not need to evaluate all switching states. Another problem of direct MPC is the increased current or torque ripple because of a poor time resolution compared to modulator based schemes. The time resolution is constrained by the minimum sampling time that is needed to solve the optimization problem. Landsmann et al. (2011) and Stolze and Karamanakos (2013) reduced the current ripple through a variable switching point that can lie between two sampling instants. Direct MPC applied to a two-mass system is presented, for example, in (Fuentes et al., 2012).

An essential part of the overall PRG structure that is presented here (see Figure 3.1 on page 40) is a stabilized closed-loop plant, or more precisely, closed-loop stability must be ensured while the system constraints are inactive. The handling of constraints is the task of the PRG that is presented in Chapter 3. The discrete-time LTI model (4.9) is used to design a model-based closed-loop control scheme. The servomechanism considered here represents closed-loop position control for a two-mass system with the following specialties compared to most other position control systems:

- The involved sampling time $T_\mathrm{s} = 10\,\mu\mathrm{s}$ is comparably low for an electrical drive system. Nevertheless, comparing T_s with the electrical time constant $\tau_\mathrm{el} \approx 60\,\mu\mathrm{s}$, which represents the smallest time constant of the system, it is desirable to lower T_s even further in order to avoid discretization problems. However, $T_\mathrm{s} = 10\,\mu\mathrm{s}$ is limited by the computation speed of the used DSP. The consequence for the PRG scheme of Chapter 3 is that the schemes presented in this thesis need to be executed offline ($\mathrm{online}_\mathrm{PRG} = \mathrm{false}$) and therefore no feedback information can be used ($\mathrm{feedback}_\mathrm{PRG} = \mathrm{false}$). The reason

for the offline execution is the time it takes to solve the involved PRG optimization problems, which is not possible in 10 µs for the DSP used here. There are, however, efforts in academia to solve these optimization problems in the µs range using field programmable gate arrays (FPGAs) (Jerez et al., 2013). These efforts and the steadily increasing computational power suggest that the algorithms presented here can be executed in real-time in the near future.

- The state $\varphi_\mathrm{l}[k]$, which should be controlled, cannot be measured in normal operation without high additional cost and effort. Hence, the load position $\varphi_\mathrm{l}[k]$ is estimated using a state estimator.

- In contrast to most electrical drive systems, the application that is treated here does not involve an external load torque. Hence, the control reserve (CR) can be kept low if a precise system model is available. This allows the PRG to nearly exploit the full dynamic range of the system. A CR represents the difference between the voltage limit v_max considered in the PRG and the "real" voltage limit v_DC of the power supply. The small CR allows the feedback controller to compensate for small disturbances (e.g., unmodeled nonlinear friction or model mismatch).

- The current dynamics and the position dynamics lie within the same range. Hence, a classical assumption of cascaded control that the dynamics of the current control loop can be neglected does not hold here. The current and position dynamics also influence the handling of constraints. There exist trajectory planning approaches that deliver reference trajectories with limited acceleration, which is proportional to the current, in order to account for current constraints. These approaches, however, assume that the current dynamics can be neglected. For the application considered here this assumption would lead to reference trajectories that cannot be tracked by the control system, as the hard voltage constraints are reached because of non-negligible current dynamics. Summing up, it is necessary to account for current *and* voltage constraints in this work.

Similar problems arise for the control of atomic force microscopes (Schitter and Astrom, 2007; Jerez et al., 2013) or for the control of hard disc drives (Guo et al., 2011). These applications also involve resonant modes, low sampling times, and constraints that have to be handled. The sampling times, which are often in the µs range, make it challenging to execute optimization-based control algorithms in real-time.

The PRG structure used in this work makes use of 2-DoF based control in order to track the optimized trajectories of the PRG. A 2-DoF controller consists of a feedback part, which makes the control scheme robust against disturbances, and a feedforward part, which is responsible for a fast transient response. The feedback controller is a linear quadratic

regulator (LQR) that requires the full state vector $\boldsymbol{x}[k]$, as it is a state-space based method. As not all states are measured, a state estimator is needed to reconstruct the complete state vector including the load position $\varphi_\mathrm{l}[k]$, which needs to be controlled but is *not* measured.

The following sections roughly describe the feedback and feedforward control design process for a single motor. The 2-DoF structure is designed separately for each motor but the control algorithms for both axes/motors run on a single DSP.

4.2.1 State Estimation

State-space based control methods make use of the *complete* state vector in order to design a suitable feedback controller. Most of the time, only a fraction of the states is measured—it is therefore necessary to use state estimation[5] to get the complete vector.

The classical state estimator for deterministic systems originates from (Luenberger, 1964) and uses the measured output, the system input, and the system model to determine the full state vector. The difference between the estimated output and the measured output (estimator error) multiplied by a gain matrix represents the feedback part in the classical Luenberger observer. The gain matrix determines how fast the estimated states converge to the "real" states in presence of disturbances or an initial state error.

The Kalman filter, invented by Kalman (1960), has the same structure as the estimator proposed by Luenberger (1964). The essential difference, however, is the method to determine the estimator gain matrix. The so-called Kalman gain is determined such that the mean square error of the estimated states is minimized in the presence of Gaussian (white) noise (process and measurement noise). The identified model in the form of the discrete-time LTI system (4.9) and the output matrix $\boldsymbol{C}$ (4.8) are the foundation for the design of the Kalman filter that is used in this work.

In order to account for unmodeled phenomena, for example nonlinear friction, the identified system model is extended in order to estimate a disturbance current $i^\mathrm{d}[k]$. This disturbance current should be able to compensate the slow components of the nonlinear friction torque, for example, Coulomb friction. The used disturbance estimator belongs to the family of estimators introduced in (Hostetter and Meditch, 1973). A survey of other available disturbance estimator schemes is presented in (Radke and Gao, 2006). Moreover, the disturbance estimator replaces an integral action in state-space control and thus avoids wind-up problems, which are common when an integral action is involved.

Further details on state estimation are omitted here, as the focus of this work lies on PRGs.

[5] Another common term—instead of *estimator*—is *observer*. The term *estimator*, however, describes its function more precisely because *observe* normally implies measurement (see Franklin et al., 1997, p. 281).

4.2.2 Two-Degrees-of-Freedom based Control

A two-degrees-of-freedom (2-DoF) based approach allows to separate basic control system specifications in terms of controller design. The feedforward part of the 2-DoF part ensures a fast dynamic response, whereas the feedback part ensures stability and robustness.

4.2.2.1 Feedback Control Design

The feedback part of the 2-DoF based control approach shall stabilize a possibly unstable plant. Furthermore, feedback introduces robustness against uncertain plant parameters, unmodeled dynamics, and disturbances.

Controller design in state-space is used here to design the feedback controller. A classical state-space controller (Franklin et al., 1997, p. 279ff.) for a general discrete-time MIMO LTI system in the form of (4.9) reads as

$$\boldsymbol{u}[k] = \boldsymbol{V}\boldsymbol{y}^{\mathrm{r,ff}}[k] - \boldsymbol{K}\boldsymbol{x}[k], \tag{4.11}$$

which leads to the following system model

$$\begin{aligned} \boldsymbol{x}[k+1] &= \underbrace{(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{K})}_{\hat{\boldsymbol{A}}}\boldsymbol{x}[k] + \underbrace{\boldsymbol{B}\boldsymbol{V}}_{\hat{\boldsymbol{B}}}\boldsymbol{y}^{\mathrm{r,ff}}[k], \\ \boldsymbol{y}[k] &= \boldsymbol{C}\boldsymbol{x}[k]. \end{aligned} \tag{4.12}$$

This closed-loop system exhibits a new input $\boldsymbol{y}^{\mathrm{r,ff}}[k] \in \mathbb{R}^{n_y}$ that represents the feedforward processed reference signal for the feedback controller (see Figure 4.5). The choice of the feedback matrix $\boldsymbol{K} \in \mathbb{R}^{n_u \times n_x}$ determines the eigenvalues of the matrix $\hat{\boldsymbol{A}} \in \mathbb{R}^{n_x \times n_x}$ and therefore the closed-loop poles of the system. The matrix $\boldsymbol{V} \in \mathbb{R}^{n_u \times n_y}$ is chosen such that the steady-state value of the control error $\boldsymbol{y}^{\mathrm{r,ff}}[k] - \boldsymbol{y}[k]$ is zero.

For simple systems with a small number of states n_x there exist simple rules to place the eigenvalues of $\hat{\boldsymbol{A}}$ (and subsequently determine $\boldsymbol{K}$) in order to achieve a desired system response. For higher order systems, the closed-loop poles can be placed, for example, by choosing dominant second-order poles in order to get a desired step response (Franklin et al., 2002, p. 530).

A popular method to determine the feedback gain $\boldsymbol{K}$ is the LQR approach (Franklin et al., 1997, pp. 371), which is also used in this work. The feedback gain $\boldsymbol{K}$ is the solution of an algebraic Riccati equation that originates from an infinite horizon optimal control problem. This optimal control problem involves weighting matrices that allow to find a trade-off between the control effort and a fast response to disturbances and reference changes. If the

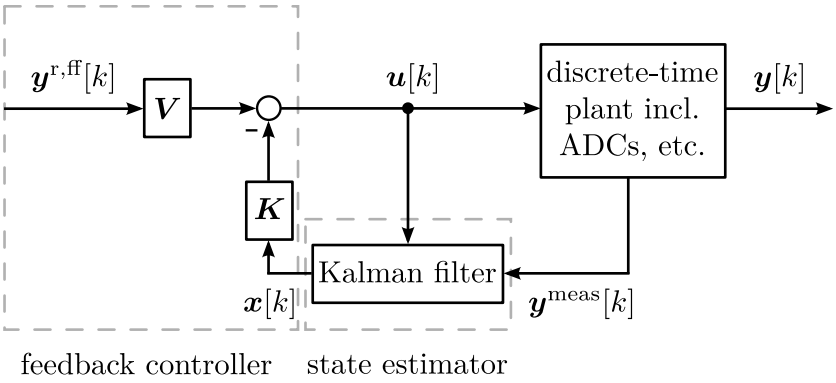


Figure 4.5: Illustration of the closed-loop controlled plant. The signal $\boldsymbol{y}^{\mathrm{r,ff}}[k]$ represents the output of a respective feedforward part. The discrete-time plant that includes the analog-to-digital converters (ADCs) and power-electronics is shown in Figure 4.3.

respective LQR weights are chosen larger than zero, assuming controllability, the closed-loop system is stable even if the open-loop system is unstable.

A single motor of the positioning system described in Section 4.1 represents a discrete-time single-input, single-output (SISO) LTI system, where $n_x = 5$, $n_u = 1$, and $n_y = 1$. In contrast to the output matrix $\boldsymbol{C}$ that is used for identification and state estimation (4.8), the output matrix used for feedback control is defined as

$$\boldsymbol{C} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{4.13}$$

which makes the load position $\varphi_{\mathrm{l}}[k]$ the output of the system. The LQR relies on the estimated state vector[6] $\boldsymbol{x}[k]$ that is provided by the Kalman filter. The operation of a Kalman filter together with an LQR leads to a so-called linear quadratic Gaussian (LQG) controller.

As discussed in Section 4.2.1, in addition to the "normal" states, a disturbance current $i^{\mathrm{d}}[k]$ is estimated. In order to compensate the disturbance, the current $i[k]$ is replaced by $i[k]-i^{\mathrm{d}}[k]$ in the estimated state vector $\boldsymbol{x}[k]$.

If the discrete-time LTI system (4.9) is unstable, the stable closed-loop model (4.12) should be used for the design of the PRG to ease the overall design process. The closed-loop system has $\boldsymbol{y}^{\mathrm{r,ff}}[k]$ as its input instead of $\boldsymbol{u}[k]$. Nevertheless, as $\boldsymbol{u}[k]$ can be considered to be a virtual system output in the form of $\boldsymbol{u}[k] = \boldsymbol{V}\boldsymbol{y}^{\mathrm{r,ff}}[k] - \boldsymbol{K}\boldsymbol{x}[k]$, it is straightforward to constrain $\boldsymbol{u}[k]$ and to determine $\boldsymbol{u}^{\mathrm{r,opt}}[k]$.

[6] Estimated states are *not* denoted by the commonly used "hat" (e.g., $\hat{\boldsymbol{x}}[k]$) in this work. Instead, measured states are marked, for example, by $\varphi_{\mathrm{m}}^{\mathrm{meas}}[k]$, whereas the corresponding estimated value reads as $\varphi_{\mathrm{m}}[k]$. Most experimental results in this work are estimated values, which is reasonable as the estimates are validated by comparing them with the respective measured values in Section 4.3.

4.2.2.2 Feedforward Control Design

The second part that belongs to a 2-DoF based control structure is the feedforward part. The aim of feedback is to stabilize and increase the robustness of the controlled system; feedforward methods aim to improve the dynamic behavior and to achieve tracking of the trajectories that are fed to the feedforward part. However, these trajectories can only be tracked if they are designed such that system constraints are maintained—especially hard constraints are critical. The first feedforward method is called dynamic, model-based feedforward control (DynFF) and relies on the optimized state $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ and input $\boldsymbol{u}^{\mathrm{r,opt}}[k]$, which are naturally available in a PRG scheme. The second method is a classical system inversion method called zero-phase-error tracking control (ZPETC), which aims to invert the TF of the closed-loop system. The fact that the plant model comprises sampling zeros that are non-minimum-phase plays an important role for system inversion.

Dynamic, Model-based Feedforward Control (DynFF) In order to track the optimized trajectory, which is delivered by the PRG, the 2-DoF controller design is completed with a feedforward approach. As the trajectories of the complete state vector $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ and the trajectory of the input $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ are naturally available as outputs of the PRG (see Figure 3.1 on page 40), it is advisable to use the DynFF concept for tracking (Roppenecker, 2009). The DynFF approach is also used, for example, in differential flatness based control (Graichen, 2006; Thomsen and Fuchs, 2011).

A 2-DoF based state-space controller with DynFF according to (Roppenecker, 2009) for a system in the form of (4.9) is written as

$$\boldsymbol{u}[k] = \boldsymbol{u}^{\mathrm{r,opt}}[k] + \boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{r,opt}}[k] - \boldsymbol{x}[k]\right), \tag{4.14}$$

where $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ is the optimized state vector and $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ is the optimized input vector. These values are outputs of the PRG. The DynFF law (4.14) is not compatible with the input $\boldsymbol{y}^{\mathrm{r,ff}}[k]$ of the feedback control scheme that is shown in Figure 4.5. Nevertheless, it is shown later that this feedback control scheme can still be used without structural modifications. Furthermore, the scheme presented in Figure 4.5 is needed for the second feedforward control concept that is presented later. The feedback gain $\boldsymbol{K}$ in (4.14) remains as designed for feedback control. The difference equation for the state error $e^{\mathrm{s}}[k] = \boldsymbol{x}^{\mathrm{r,opt}}[k] - \boldsymbol{x}[k]$ is given, combining (4.14) and (4.9), as

$$e^{\mathrm{s}}[k+1] = \underbrace{(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{K})}_{\hat{\boldsymbol{A}}} e^{\mathrm{s}}[k] + \boldsymbol{x}^{\mathrm{r,opt}}[k+1] - \left(\boldsymbol{A}\boldsymbol{x}^{\mathrm{r,opt}}[k] + \boldsymbol{B}\boldsymbol{u}^{\mathrm{r,opt}}[k]\right). \tag{4.15}$$

As $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ and $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ are determined by the PRG to fulfill

$$\boldsymbol{x}^{\mathrm{r,opt}}[k+1] = \boldsymbol{A}\boldsymbol{x}^{\mathrm{r,opt}}[k] + \boldsymbol{B}\boldsymbol{u}^{\mathrm{r,opt}}[k], \tag{4.16}$$

the state error difference equation (4.15) reduces to

$$e^{\mathrm{s}}[k+1] = \hat{\boldsymbol{A}} e^{\mathrm{s}}[k]. \tag{4.17}$$

This error difference equation shows that the controller only controls an initial state error $e^{\mathrm{s}}[0] = \boldsymbol{x}^{\mathrm{r,opt}}[0] - \boldsymbol{x}[0]$ to zero if (4.16) is satisfied and no disturbances are present. This means that the feedback controller is inactive once $e^{\mathrm{s}}[k] = 0$ is reached—even in dynamic operation, which implies perfect tracking of the reference. These positive properties are attenuated in presence of model mismatch or unmeasurable disturbances. In both cases, the feedback controller gets active.

The closed-loop feedback controlled system (4.12) in Section 4.2.2.1 has $\boldsymbol{y}^{\mathrm{r,ff}}$ as its input. This input, which is the output of the DynFF part, can be determined by using (4.14) and comparing it with the standard feedback control law (4.11) as

$$\boldsymbol{y}^{\mathrm{r,ff}}[k] = \boldsymbol{V}^{-1}\left(\boldsymbol{u}^{\mathrm{r,opt}}[k] + \boldsymbol{K}\boldsymbol{x}^{\mathrm{r,opt}}[k]\right), \tag{4.18}$$

if $\boldsymbol{V}$ is a square matrix ($n_y = n_u$). Here, for the SISO case, $\boldsymbol{V}$ is a scalar value. The control structure can be slightly changed to allow a non-square $\boldsymbol{V}$ (see Roppenecker, 2009).

Zero-Phase-Error Tracking Controller (ZPETC) The second feedforward concept, which is presented here, was introduced by Tomizuka (1987) and is called ZPETC. This feedforward concept is also model-based but relies—in contrast to DynFF—on system inversion. System inversion means in principle that the poles and zeros of the *closed-loop* TF are compensated by the respective poles and zeros of the inverse TF (feedforward). Therefore, ideally, the poles of the closed-loop TF become the zeros of the feedforward TF and vice versa. The exact location of the closed-loop poles is determined by the tuning of the LQR. Nevertheless, they lie within the unit-circle of the discrete-time complex plane and are therefore stable—this leads to minimum-phase zeros in the inverse TF. The closed-loop zeros, however, are not influenced by the LQR and therefore the non-minimum-phase zeros (sampling zeros) remain. The closed-loop TF of the application considered here comprises two non-minimum-phase zeros. These zeros cannot be canceled by the respective poles, as the inverse TF would get unstable. The ZPETC approach handles these non-minimum-phase zeros such that the overall (2-DoF) TF has zero phase delay (see Tomizuka, 1987, for details). Furthermore, the ZPETC approach ensures unity gain in the low and medium frequency regions. If the

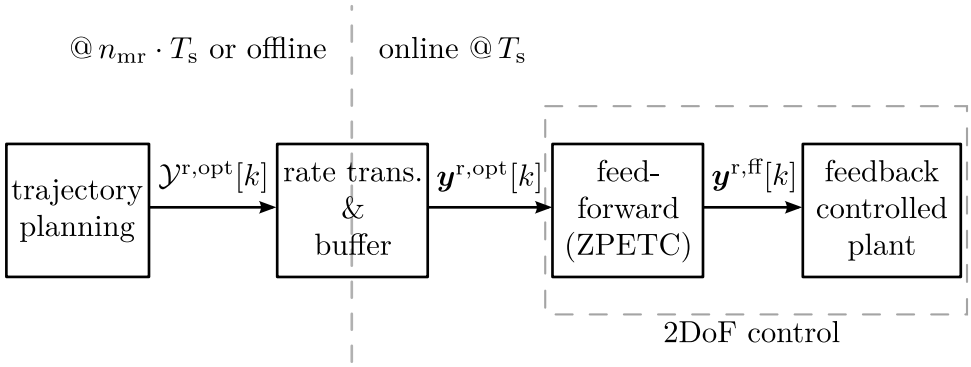


Figure 4.6: Detailed trajectory tracking structure including the zero-phase-error tracking control (ZPETC) feedforward part

resulting ZPETC based feedforward TF is non-causal, simple unit delays are used to make it causal, which leads to a phase delay that is *non-zero*. This leads for the plant treated here to a constant phase delay of $2 \cdot T_{\mathrm{s}}$. A ZPETC based feedforward path is used in this work to track trajectories for which only the optimized reference $\boldsymbol{y}^{\mathrm{r,opt}}[k]$ is available—and not the complete state $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ and input $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ as is the case for DynFF. The closed-loop plant together with ZPETC does *not* have unity gain for high frequencies, therefore—compared to the DynFF approach—a tracking error is present for references with a considerable high-frequency content. Other system inversion techniques for non-minimum-phase systems are discussed and compared to the ZPETC approach in (Butterworth et al., 2008).

The detailed control and trajectory planning structure that relies on ZPETC based feedforward control is shown in Figure 4.6. It is a multi-rate scheme analog to the PRG scheme (see Figure 3.1 on page 40). The overall scheme presented in Figure 4.6 is considered in this work to be an advanced industry standard and is used for comparison with the PRG schemes. The main difference to the PRG approach is that the trajectory planning part directly outputs $\boldsymbol{y}^{\mathrm{r,opt}}[k]$, or more precisely $\mathcal{Y}^{\mathrm{r,opt}}[k]$, as there is no PRG used in this approach. The lack of a PRG causes the need of a more "intelligent" trajectory planning part. This means that, if no PRG is used, the trajectory planning part has to handle constraints in order to ensure stability of the overall system. For example, simple references in the shape of a step or a ramp may become problematic. A trajectory planning method, which is de facto standard in industry, is presented in Section 5.3.2. The whole scheme is considered to be an *advanced* industry standard because a single model-based controller is used here in contrast to the mainly used cascaded structure.

4.3 Validation of the Estimated Load Position

In this section, experimental results are shown, which confirm that it is reasonable to omit a cost-intensive, or even impractical sensor for the load position. Two steps are necessary to control the load position without a sensor. First, the plant has to be modeled and identified using the underlying physics in order to preserve the physical connection between states—the sole identification of the input-output behavior is not enough (see Section 4.1). The second step is the use of a state estimator that delivers estimates of the plant states, including the load position $\varphi_\mathrm{l}[k]$, which should be controlled (see Section 4.2.1).

In order to measure the load positions of both axes ($\varphi_{\mathrm{l},1}^{\mathrm{meas}}[k]$ and $\varphi_{\mathrm{l},2}^{\mathrm{meas}}[k]$) for validation purposes, the experimental setup illustrated in Figure 4.1 on page 46 was used. A position sensitive device (PSD) allows measuring the position of the laser spot in two dimensions. The position measured by the PSD is dependent on the load positions of both PM DC motors by a nonlinear geometric position transformation (Tang et al., 2004). For the application considered here, a small angle approximation that leads to a linear relationship between the measured PSD positions and the load positions of the motors is reasonable. Hence, after a calibration process including two offsets and two gain values, the PSD delivers the measured load angles $\varphi_{\mathrm{l},1}^{\mathrm{meas}}[k]$ and $\varphi_{\mathrm{l},2}^{\mathrm{meas}}[k]$. The experimental setup illustrates why a measurement of the spot position might not be possible in normal operation—often, a workpiece has to be placed at the position of the PSD.

The following estimation errors are defined to quantify the estimation quality

$$\begin{aligned} e_{\mathrm{m},1}^{\mathrm{est}}[k] &= \varphi_{\mathrm{m},1}^{\mathrm{meas}}[k] - \varphi_{\mathrm{m},1}[k], \\ e_{\mathrm{m},2}^{\mathrm{est}}[k] &= \varphi_{\mathrm{m},2}^{\mathrm{meas}}[k] - \varphi_{\mathrm{m},2}[k], \\ e_{\mathrm{l},1}^{\mathrm{est}}[k] &= \varphi_{\mathrm{l},1}^{\mathrm{meas}}[k] - \varphi_{\mathrm{l},1}[k], \\ e_{\mathrm{l},2}^{\mathrm{est}}[k] &= \varphi_{\mathrm{l},2}^{\mathrm{meas}}[k] - \varphi_{\mathrm{l},2}[k]. \end{aligned} \tag{4.19}$$

To cover the relevant range of systems dynamics, linearly increasing chirp signals (frequency sweep) ranging from 0 to 22 kHz are used as the references $y_1^{\mathrm{r,ff}}[k]$ and $y_2^{\mathrm{r,ff}}[k]$ of the feedback controllers. The feedforward paths are not used in order to prevent the hard input constraint of $\pm v_{\max}$ from getting active. It turned out that the chirp signals cause actuator saturation even for relatively low amplitudes if the feedforward path is used. As tracking is not the objective here, the feedforward path is omitted and the amplitude of the chirp signals was chosen such that the actuator constraints do not get active.

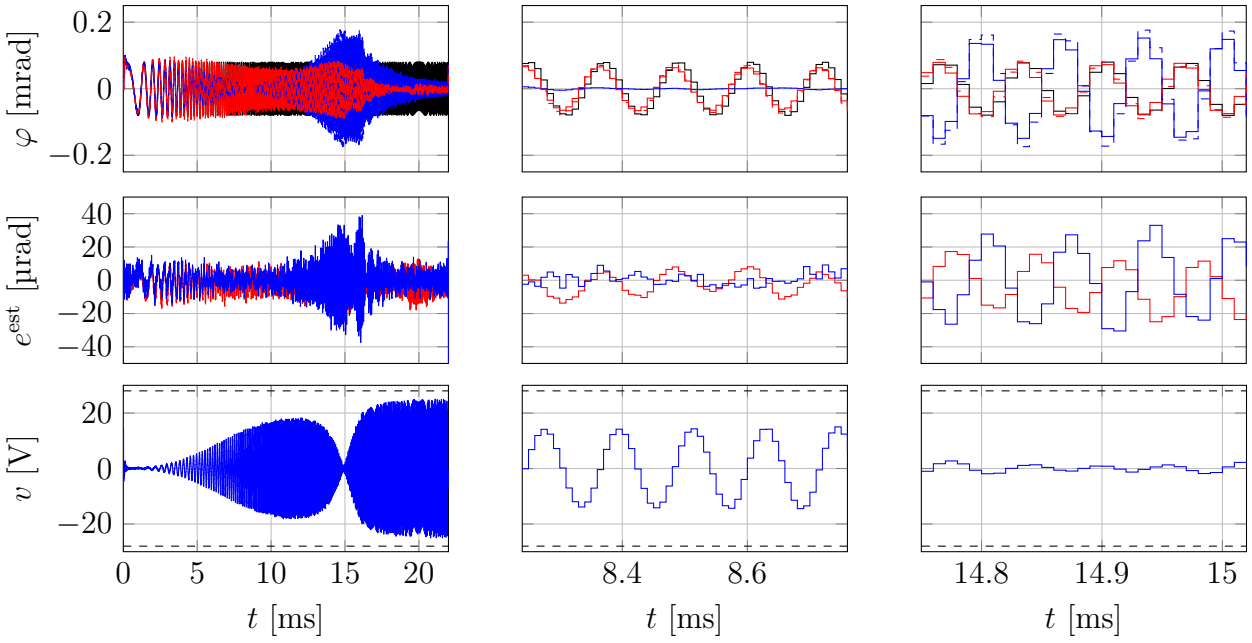
Figure 4.7 on page 66 shows the chirp signal $y_1^{\mathrm{r,ff}}[k]$, the resulting motor positions $\varphi_{\mathrm{m},1}[k]$ and $\varphi_{\mathrm{m},1}^{\mathrm{meas}}[k]$, as well as the resulting load positions $\varphi_{\mathrm{l},1}[k]$ and $\varphi_{\mathrm{l},1}^{\mathrm{meas}}[k]$ (including zoom-in plots for the anti-resonant and resonant frequencies). The motor position estimation

error $e_{\mathrm{m},1}^{\mathrm{est}}[k]$ and the load position estimation error $e_{\mathrm{l},1}^{\mathrm{est}}[k]$ are shown in the middle plot in order to quantify the errors between the estimated and the measured values. Over a large frequency range, the estimation errors lie within about $\pm 10\,\mu\mathrm{rad}$, except in the range of the resonance frequency, where the errors lie within about $\pm 30\,\mu\mathrm{rad}$. The resolution of the motor position sensor—considering the quantization of the analog-to-digital converter (ADC)—is about $1\,\mu\mathrm{rad}$. The analysis of the noise (with a disabled feedback controller), however, led to a standard deviation of the noise of about $4.9\,\mu\mathrm{rad}$. Hence, this standard deviation mainly determines the accuracy of the position sensor. Comparing the noise level of the sensor to the estimation errors (especially to the load position error), it is confirmed that it is reasonable to omit a load position sensor and to rely on the *estimated* load position $\varphi_{\mathrm{l},1}[k]$. The estimation errors that are comparably low but still above the noise level might result from unmodeled dynamics, model mismatch, nonlinear friction, the small angle approximation of the optical path, and the assumption that the torque constant does not vary with the angular position.

The estimation errors are almost identical in the low-frequency region, where the system behaves like a one-mass system, which means that the motor and the load position are identical. The estimation errors of about $\pm 15\,\mu\mathrm{rad}$ in the low-frequency range probably result from nonlinear friction. An improved disturbance model might reduce these low-frequency estimation errors. In the mid-frequency region, where the motor position and the load position start to differ, both estimation errors are in the range of $\pm 10\,\mu\mathrm{rad}$—an error in this range is considered to be sufficiently small. The high-frequency range around the resonance frequency exhibits the highest estimation errors of about $\pm 30\,\mu\mathrm{rad}$. The reason for this might be a slight mismatch of the damping d and of the spring constant c. Furthermore, high motor and load speeds that are reached in the high-frequency range might be another reason for the comparably high estimation errors, as modeling errors might lead to speed dependent estimation errors. Nevertheless, considering the relatively high frequency, the estimation errors are still satisfactory.

The zoom-in plots on the anti-resonant and resonant frequencies show two extreme cases. The signals around the anti-resonant frequency show that the load position follows the reference quite well, although the signal of the motor position approaches zero[7]. The other extreme case is the frequency range around the resonance frequency. Again, the reference signal $y_1^{\mathrm{r,ff}}[k]$ is followed, due to the feedback controller dynamics, quite well by the load position $\varphi_{\mathrm{l},1}[k]$. The amplitude of the motor position $\varphi_{\mathrm{m},1}[k]$ is higher than the amplitude of $\varphi_{\mathrm{l},1}[k]$ and the signals show a phase shift of about $180°$. The voltage $v_1[k]$ approaches zero in the region of the resonant frequency—this is an interesting aspect in consideration of power

[7] Traditional control methods that neglect the two-mass system cannot achieve this behavior.



(a) Complete frequency range

(b) Zoom-in on the anti-resonant frequency

(c) Zoom-in on the resonant frequency

Figure 4.7: Axis 1 load position validation: A linearly increasing chirp signal from 0–22 kHz excites all relevant system dynamics (e.g., 10 kHz $\widehat{=}$ 10 ms).

Top plot: Reference signal of the feedback controller $y_1^{\mathrm{r,ff}}[k]$ (——), estimated motor position $\varphi_{\mathrm{m},1}[k]$ (——), measured motor position $\varphi_{\mathrm{m},1}^{\mathrm{meas}}[k]$ (- - - -), estimated load position $\varphi_{\mathrm{l},1}[k]$ (——), measured load position $\varphi_{\mathrm{l},1}^{\mathrm{meas}}[k]$ (- - - -).

Middle plot: Motor estimation error $e_{\mathrm{m},1}^{\mathrm{est}}[k]$ (——), load estimation error $e_{\mathrm{l},1}^{\mathrm{est}}[k]$ (——).

Bottom plot: Voltage $v_1[k]$ (——), maximum/minimum voltage $\pm v_{\max}$ (- - - -)

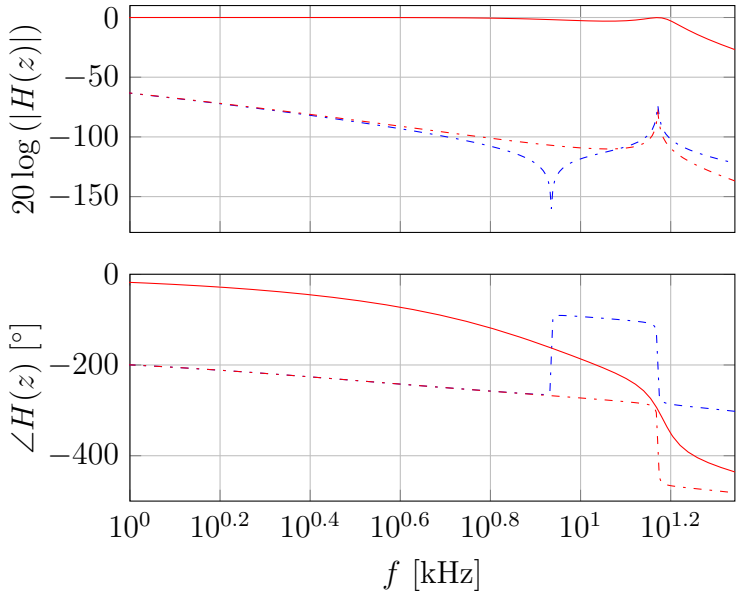
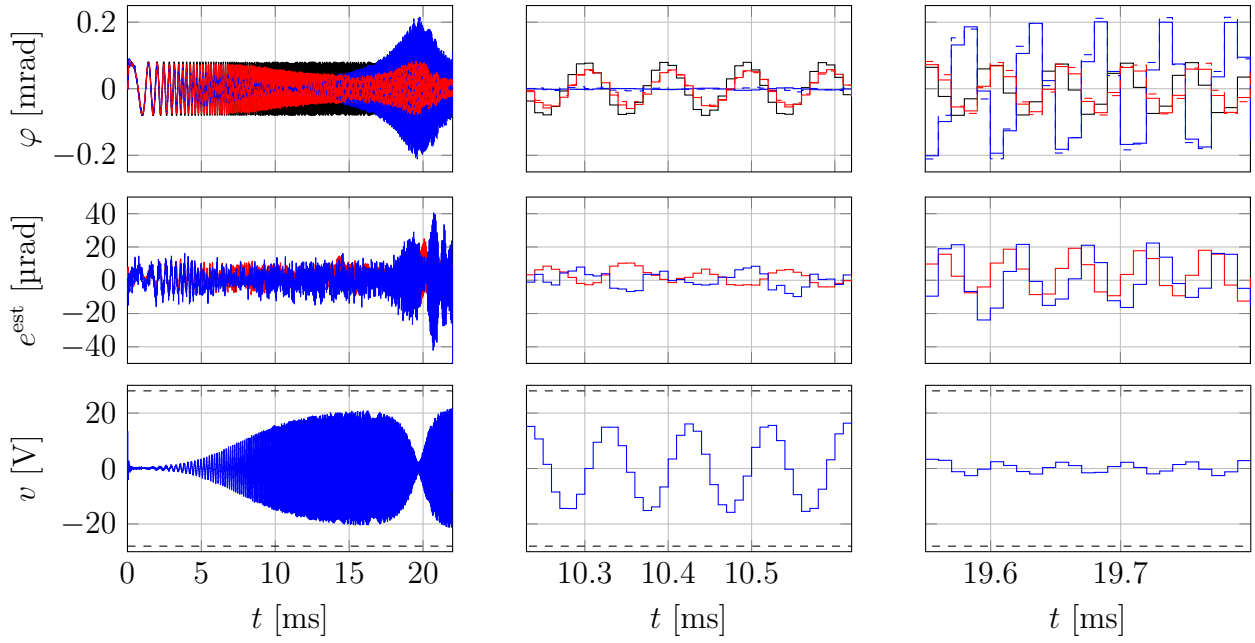


Figure 4.8: Axis 1 Bode plot: Closed-loop TF (LQR design without feedforward) for the load position $H_{y_1^{\mathrm{r,ff}}\to\varphi_{\mathrm{l},1}}(z)$ (——), open-loop TF for the motor position $H_{v_1\to\varphi_{\mathrm{m},1}}(z)$ (-·-·-), open-loop TF for the load position $H_{v_1\to\varphi_{\mathrm{l},1}}(z)$ (-··-··-)



(a) Complete frequency range

(b) Zoom-in on the anti-resonant frequency

(c) Zoom-in on the resonant frequency

Figure 4.9: Axis 2 load position validation: A linearly increasing chirp signal from 0–22 kHz excites all relevant system dynamics (e.g., 10 kHz $\widehat{=}$ 10 ms).

Top plot: Reference signal of the feedback controller $y_2^{\mathrm{r,ff}}[k]$ (——), estimated motor position $\varphi_{\mathrm{m},2}[k]$ (——), measured motor position $\varphi_{\mathrm{m},2}^{\mathrm{meas}}[k]$ (- - - -), estimated load position $\varphi_{\mathrm{l},2}[k]$ (——), measured load position $\varphi_{\mathrm{l},2}^{\mathrm{meas}}[k]$ (- - - -).

Middle plot: Motor estimation error $e_{\mathrm{m},2}^{\mathrm{est}}[k]$ (——), load estimation error $e_{\mathrm{l},2}^{\mathrm{est}}[k]$ (——).

Bottom plot: Voltage $v_2[k]$ (——), maximum/minimum voltage $\pm v_{\max}$ (- - - -)

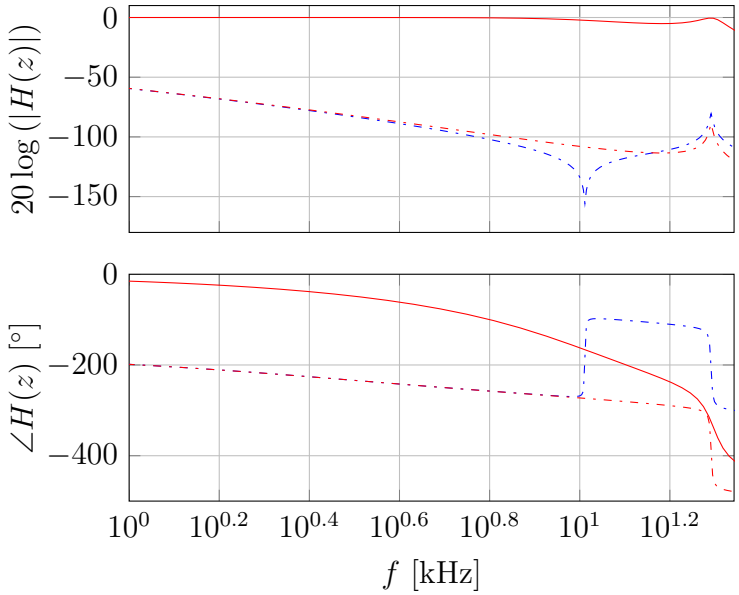


Figure 4.10: Axis 2 Bode plot: Closed-loop TF (LQR design without feedforward) for the load position $H_{y_2^{\mathrm{r,ff}} \to \varphi_{\mathrm{l},2}}(z)$ (——), open-loop TF for the motor position $H_{v_2 \to \varphi_{\mathrm{m},2}}(z)$ (-·-·-), open-loop TF for the load position $H_{v_2 \to \varphi_{\mathrm{l},2}}(z)$ (-··-··-)

losses, especially for applications that require to follow a reference with a fixed frequency and allow to use the resonant frequency of the positioning system for this purpose.

The main observations of the time-domain plots of Figure 4.7 on page 66 can also be made in the Bode plot presented in Figure 4.8 on page 66. Three different TFs are shown, namely, the closed-loop TF without feedforward $H_{y_1^{\mathrm{r,ff}} \to \varphi_{\mathrm{l},1}}(z)$, the open-loop TF with the motor position as the output $H_{v_1 \to \varphi_{\mathrm{m},1}}(z)$, and the load position open-loop TF $H_{v_1 \to \varphi_{\mathrm{l},1}}(z)$. These TFs originate from the model with identified parameters. The characteristic properties of a two-mass system are visible in $H_{v_1 \to \varphi_{\mathrm{m},1}}(z)$ and $H_{v_1 \to \varphi_{\mathrm{l},1}}(z)$. In the range of the anti-resonant frequency, the amplitudes of the load and motor positions significantly differ. The amplitudes do not differ that much in the range of the resonant frequency, however, the overall gain is relatively high and the motor and load positions are about 180° phase shifted. The closed-loop TF (without feedforward) $H_{y_1^{\mathrm{r,ff}} \to \varphi_{\mathrm{l},1}}(z)$ originates from the LQR design and exhibits unity gain over a large frequency range which is also visible in the time-domain plots.

The results discussed up to now originate from axis/motor 1, the corresponding results for axis/motor 2 are shown in Figures 4.9 and 4.10 on page 67. The basic insights remain the same—the characteristic two-mass frequencies, however, are shifted to a higher frequency range because of a different ratio of the load/motor moments of inertia (see identified plant parameters in Table 4.1 on page 55).

Summing up, the experimental results show estimation errors that are only slightly above the noise level in the frequency range of interest. Hence, the model comprising the identified parameters is precise enough to rely on the estimated load position for control. Following this reasoning, the experimental results, which are presented in the following chapters, show estimated values. The use of estimated values allows to show the complete state vector.

5 Near Minimum-Time PRGs for Constrained SISO LTI Systems

This chapter presents an application of the predictive reference governor (PRG) approach that is introduced in Chapter 3. Two PRG approaches with different cost functions are analyzed and compared. The first PRG relies on the standard model predictive control (MPC) cost function (quadratic form) of Section 2.2. It is shown with a simple point-to-point positioning example that a quadratic form based cost function can exhibit a large overshoot in the step response. This is especially the case when the weights are tuned to get a fast response. In order to reduce this overshoot, which might be critical in some applications that require precise positioning, a PRG based on the ℓ_1-norm (sum of absolute values) is introduced. A PRG with an ℓ_1-norm based cost function shows—compared with a quadratic form based PRG—a step response where the reference value is reached in near minimum-time and with a reduced overshoot. The proposition that the reference value is reached in near minimum-time is validated for various step heights by comparing the ℓ_1-norm based step response to the minimum-time solution that is gained through optimal control. To complete the comparison, the two PRGs and the minimum-time solution are compared to an advanced industry standard, which is based on acceleration- and jerk-limited trajectories.

The proposed near minimum-time PRG formulation aims to exploit the constraints. In other words, the constraints should not just be respected but rather fully exploited to achieve a fast transient response. A near minimum-time response is achieved by an ℓ_1-norm based cost function and the respective tuning of the involved weighting factors.

Due to the low sampling time of $T_\mathrm{s} = 10\,\mu\mathrm{s}$ and the relatively large prediction horizon, the two PRG approaches have to be executed offline ($\mathrm{online}_\mathrm{PRG} = \mathrm{false}$) and therefore no feedback information can be used ($\mathrm{feedback}_\mathrm{PRG} = \mathrm{false}$). Hence, the behavior of the plant is simulated in order to propagate the receding horizon of the PRGs over the reference trajectory.

5.1 Standard PRG Formulation

The here called standard PRG is based on the PRG scheme of Chapter 3, the underlying optimization problem is the standard MPC formulation of Section 2.2. This optimization problem consists of a cost function based on a quadratic form together with input and state constraints.

More specific, the MPC optimization problem (2.32) that represents a quadratic program (QP) is solved in a receding horizon manner according to the PRG Algorithm 1, which is presented in Chapter 3. The underlying cost function of the optimization problem (2.32) is repeated here for reasons of completeness

$$J_{\mathrm{q}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}}\, \boldsymbol{E}^{\mathrm{r}}[k] \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}}\, \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2 . \tag{5.1}$$

The two objectives of the cost function (5.1) are the control error $\boldsymbol{E}^{\mathrm{r}}[k]$ and the rate of input change $\boldsymbol{\Delta U}[k]$. These two terms are weighted with the weighting matrices $\boldsymbol{Q}_{\mathrm{q}}$ and $\boldsymbol{R}_{\mathrm{q}}$.

A general aim of this work is to fully exploit the constraints in order to get a fast control result, which means that the weighting of the control error needs to be much higher than the weighting of the rate of input change. If the rate of input change is omitted in the cost function, it is *not* guaranteed that the resulting optimization problem is convex (see Section 2.2.3 for details). Furthermore, even a small weight on the rate of input change can attenuate high-frequency content in the input signal with only a marginal effect on the overall control performance. An attenuated high-frequency content often plays a role when unmodeled higher-order dynamics (e.g., further resonances) are present.

PRG approaches based on quadratic cost functions are presented in (Stoican et al., 2012; Aghaei et al., 2013). These approaches, however, do not focus on minimum-time responses.

A cost function based on a quadratic form weights large control errors much higher than small values. Speaking in terms of a reference step, the large control error that is present at the beginning dominates the overall cost function value. This predominance leads to a fast rise time in order to quickly decrease the value of the quadratic cost function and a subsequent overshoot. The non-uniform control error weighting of a quadratic form based cost function undervalues the comparably small control errors of an overshoot compared with the large control errors at the beginning of a step. These propositions are validated by experimental results that are presented in Section 5.3.4. As there are many applications that require a small or even no overshoot (e.g., in precise positioning) together with a fast settling time, this drawback of quadratic form based PRGs is tackled in the next section.

5.2 Near Minimum-Time PRG Formulation

The presented standard PRG scheme is adapted in this section to achieve near minimum-time control results. It is proposed to substitute the quadratic form based cost function by an ℓ_1-norm based cost function. Compared to a quadratic form based cost function, the ℓ_1-norm formulation ensures a uniform weighting of the cost function values, which efficiently decreases the overshoot in the step response. Furthermore, experimental results, which are presented later on, show that the step response of an ℓ_1-norm PRG only marginally differs from the minimum-time response gained through optimal control. Compared to the standard PRG formulation, which can be cast as a QP, an ℓ_1-norm based optimization problem can be cast as a linear program (LP) (see Section 2.1.5.2). An LP requires a computationally less complex solver than a QP.

It is interesting to note that historically MPC, which is the foundation of PRGs, first started with LPs in 1963 (Propoi, 1963). Nowadays, MPC based on QPs is the de facto standard—LPs are rarely used, mostly to reduce the computational complexity of MPC algorithms that originally rely on a quadratic form (Dave et al., 1997; Stumper et al., 2012). LPs can also originate from ℓ_1-norm or ℓ_∞-norm based cost functions, for example. Bemporad et al. (2002a) present ℓ_1-norm and ℓ_∞-norm based MPC approaches—the focus, however, does not lie on a minimum-time response. Van den Broeck et al. (2010) introduced a predictive reference prefilter based on the ℓ_∞-norm to achieve near minimum-time results—mainly for step references and not for arbitrary reference signals.

In this work, the advantage of a reduced computational complexity of a solver for LPs is combined with a smaller overshoot and a near minimum-time control behavior through a PRG based on an ℓ_1-norm formulation. However, Rao and Rawlings (2000) showed that this norm introduces some drawbacks: The ℓ_1-norm can introduce idle and deadbeat control behavior. Idle control means that even though a non-zero control error is present, the solution of the ℓ_1-norm based optimization problem is zero control action (see Rao and Rawlings, 2000, for illustrative examples). Penalizing the rate of control input change $\boldsymbol{\Delta U}[k]$ instead of the often used absolute value $\boldsymbol{U}[k]$ can reduce the size of the regions in state-space where idle and deadbeat control occur (see Saffer and Doyle, 2004, for details). Another reason for choosing $\boldsymbol{\Delta U}[k]$ instead of $\boldsymbol{U}[k]$ is the intention to achieve an improved dynamic control behavior. Penalizing $\boldsymbol{\Delta U}[k]$ influences the rate of input change and therefore the frequency content of the resulting input but not the *absolute* value of the input. Furthermore, penalizing $\boldsymbol{\Delta U}[k]$ allows large absolute values of the input over relatively long periods of time, which are necessary to increase the controller dynamics. The general tendency of an ℓ_1-norm based optimization problem to deadbeat control is not necessarily a drawback, it can lead to near minimum-time control results, as deadbeat and minimum-time control are

closely related (Crossley and Porter, 1974). This feature is exploited in this work to achieve near minimum-time control results. Nevertheless, deadbeat behavior can be a drawback in the presence of noise and when the system output is close to the reference—noise can lead to "nervous" control behavior in this case. However, in the offline variant of the PRG there is no noise fed to the PRG, as there is no feedback. The problem of deadbeat control will require further attention if a PRG is operated with feedback.

In contrast to the standard PRG formulation of Section 5.1, the near minimum-time PRG formulation is presented in more detail in the following, as the concepts have not yet been presented in a previous section. The cost function based on the ℓ_1-norm consists of the same terms as the quadratic form based cost function (5.1). It is given as

$$J_{\ell_1}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} \boldsymbol{Q}_{\ell_1} \boldsymbol{E}^{\mathrm{r}}[k] \\ \boldsymbol{R}_{\ell_1} \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_1 . \tag{5.2}$$

This cost function consists of the future control error $\boldsymbol{E}^{\mathrm{r}}[k]$, which is weighted with the diagonal matrix $\boldsymbol{Q}_{\ell_1}$, and the rate of input change $\boldsymbol{\Delta U}[k]$, which is weighted with the diagonal matrix $\boldsymbol{R}_{\ell_1}$. Hence, the resulting control performance is a trade-off between the minimization of the future control error $\boldsymbol{E}^{\mathrm{r}}[k]$ and the minimization of $\boldsymbol{\Delta U}[k]$.

The weighting matrices are defined analogously to the quadratic case as

$$\begin{aligned} \boldsymbol{Q}_{\ell_1} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{q}_{\ell_1}\} \in \mathbb{R}^{N \cdot n_y \times N \cdot n_y}, \\ \boldsymbol{R}_{\ell_1} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\ell_1}\} \in \mathbb{R}^{N \cdot n_u \times N \cdot n_u}, \end{aligned} \tag{5.3}$$

where $\boldsymbol{q}_{\ell_1} \in \mathbb{R}_+^{n_y}$ and $\boldsymbol{r}_{\ell_1} \in \mathbb{R}_+^{n_u}$ are the weighting vectors for the control error and the rate of input change, respectively.

The cost function (5.2) is expressed depending on the optimization variable $\boldsymbol{U}[k]$ as

$$J_{\ell_1}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \overset{\substack{(2.17)\\(2.19)}}{=} \left\| \underbrace{\begin{bmatrix} \boldsymbol{Q}_{\ell_1} \mathcal{B}_y \\ \boldsymbol{R}_{\ell_1} \mathcal{B}_\Delta \end{bmatrix}}_{\boldsymbol{F}_{\ell_1}} \boldsymbol{U}[k] + \underbrace{\begin{bmatrix} \boldsymbol{Q}_{\ell_1} \left(\mathcal{A}_y \boldsymbol{x}[k] - \boldsymbol{Y}^{\mathrm{r}}[k] \right) \\ \boldsymbol{0}_{N \cdot n_u} \end{bmatrix}}_{\boldsymbol{g}_{\ell_1}[k]} \right\|_1 . \tag{5.4}$$

This cost function is a convex function, as the ℓ_1-norm of an affine and therefore convex function (here $\boldsymbol{F}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}_{\ell_1}[k]$) is again convex (Boyd and Vandenberghe, 2004, p. 24).

Bringing the input and state constraints (2.25) together with the cost function (5.4), the optimization problem reads as

$$\begin{aligned} \underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \left\| \boldsymbol{F}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}_{\ell_1}[k] \right\|_1 \\ \text{subject to} \quad & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{5.5}$$

This constrained ℓ_1-norm minimization problem can be cast as the LP

$$\begin{aligned} \underset{\boldsymbol{\beta},\, \boldsymbol{U}[k]}{\text{minimize}} \quad & \mathbf{1}^\top \boldsymbol{\beta} \\ \text{subject to} \quad & -\boldsymbol{\beta} \leq \boldsymbol{F}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}_{\ell_1}[k] \leq \boldsymbol{\beta}, \\ & \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{5.6}$$

This transformation to an LP introduces the vector $\boldsymbol{\beta}$, which is now part of the optimization variables. Details about this transformation and the final transformation to an LP in standard form are given in Section 2.1.5.2.

A solver for LPs is in general computationally less complex than a solver for QPs. The transformation of (5.5) to (5.6), however, introduces additional inequality constraints. Therefore, the time needed to solve the LP (5.6) is not necessarily shorter than the time needed to solve the corresponding QP based optimization problem of Section 5.1. The transformation of the LP (5.6) to its dual problem, as proposed in (Camacho and Bordons, 2004, p. 203), could lead to an improved computation time. A first investigation, however, did not confirm this for the application that is treated in this work.

5.3 Example: Point-to-Point Positioning of a PM DC Motor

This section validates the standard PRG approach and the proposed near minimum-time PRG for a point-to-point positioning application using a single axis of the positioning system that is presented in Chapter 4. The simple point-to-point position control example (step reference signals) allows to focus on the timing of the different approaches and on some common performance indicators, namely, overshoot (OS), rise time (RT), and settling time (ST). In order to validate the proposed near minimum-time PRG formulation, a minimum-time optimal control problem is solved to get the fastest step response that is *physically* possible. A traditional minimum-time optimal control problem deals with initial and final states and *not* with *arbitrary* varying reference signals, which can be handled by the PRG approaches. This is another reason why this chapter focuses on step responses; various other

reference signal shapes are shown later on. Furthermore, to complete the comparison, a trajectory planning method is presented that constrains the acceleration and the jerk of the reference position. This trajectory planning method together with the two-degrees-of-freedom (2-DoF) based controller is considered to be an advanced industry standard.

Additional contents of this section are considerations about the tuning of the PRG approaches and an estimation of the calculation times of the involved optimization problem solvers.

5.3.1 Tuning of the PRG Approaches

The ℓ_1-norm and the quadratic form based approaches require the tuning of weights. For the single-input, single-output (SISO) system considered here the number of tuning parameters reduces to a single parameter for each PRG approach, as the control error weights can be fixed as the scalar[1] values $\boldsymbol{q}_\mathrm{q} = \left[1\,\mathrm{rad}^{-2}\right]$ and $\boldsymbol{q}_{\ell_1} = \left[1\,\mathrm{rad}^{-1}\right]$. The remaining weights, namely the rate of input change weights $\boldsymbol{r}_\mathrm{q}$ and $\boldsymbol{r}_{\ell_1}$, are tuned to achieve comparable, fast RTs that lead to fully exploited constraints and to achieve similar times for the damping of induced oscillations of the two-mass system for the standard and the near minimum-time PRG (see Table 5.1). The fact that a *single* tuning parameter is enough makes the tuning comparably simple (keeping in mind that tuning might not be an intuitive task, i.e., non-convex). This tuning is done for a nominal reference step height of $\varphi_\mathrm{l}^\mathrm{r} = 0.5\,\mathrm{mrad}$ and is kept constant for other step heights.

Another essential parameter for PRG schemes is the prediction horizon N. As discussed in Section 3, a long prediction horizon ensures the optimality of open- *and* closed-loop trajectories. To achieve optimality, a prediction horizon that covers the whole settling time is necessary. The prediction horizon $N = 20$ is chosen here for a worst-case reference step height. In addition, a large prediction horizon emulates infinite horizon control (Mayne et al., 2000) and therefore contributes to the stability of the PRG.

5.3.2 Acceleration- and Jerk-limited Trajectory Planning: An Advanced Industry Standard

In order to classify the PRG approaches concerning constraint handling and time-optimality, a trajectory planning method that is used in industry is presented in this section. Figure 4.6 on page 63 presents the structure consisting of a trajectory planning part and a 2-DoF controlled plant (see Section 4.2.2.2). Due to the lack of a PRG, which handles constraints,

[1] The notation introduced before treats a general multiple-input, multiple-output (MIMO) system. When dealing with SISO systems in this thesis the notation is not changed. To keep the notation consistent, a scalar value is written in bold as a one-dimensional vector.

Maximum voltage $v_{\max}$	$28\,\mathrm{V}$
Maximum current $i_{\max}$	$5\,\mathrm{A}$
$\mathrm{online}_{\mathrm{PRG}}$	false
$\mathrm{feedback}_{\mathrm{PRG}}$	false
preview (reference known in advance)	false
Prediction horizon N	20
Receding horizon shift N_{shift}	1
Standard PRG	
Control error weight $\boldsymbol{q}_{\mathrm{q}}$	$\left[1\,\mathrm{rad}^{-2}\right]$
Rate of input change weight $\boldsymbol{r}_{\mathrm{q}}$	$\left[(0.00005)^2\ \mathrm{V}^{-2}\right]$
Near minimum-time PRG	
Control error weight $\boldsymbol{q}_{\ell_1}$	$\left[1\,\mathrm{rad}^{-1}\right]$
Rate of input change weight $\boldsymbol{r}_{\ell_1}$	$\left[0.0002\,\mathrm{V}^{-1}\right]$

Table 5.1: PRG parameters and system constraints of the SISO point-to-point positioning example

the trajectory planning part has to handle constraints now. A common approach is to limit the acceleration and the jerk (derivative of the acceleration) of a trajectory. Examples for such trajectory planning approaches in academia can be found in (Kröger, 2010; Lambrechts et al., 2005; Macfarlane and Croft, 2003; Erkorkmaz and Altintas, 2001). Nevertheless, acceleration- and jerk-limited trajectory planning is also widely used in industry. The PLCopen® Motion Library (Van der Wal, 2001) is non-proprietary and can be used in programmable logic controllers (PLCs) of leading companies in the field of motion control (e.g., ABB, Beckhoff, B&R, Siemens, etc.). Hence, acceleration- and jerk-limited trajectories are considered to be industry standard. The overall scheme is an *advanced* industry standard, as the 2-DoF state-space based control structure used in this work differs from the common cascaded control schemes with velocity and acceleration feedforward.

In the following, the choice of the maximum acceleration and the maximum jerk for the application considered here is discussed. Some simplifications of the state-space model (see (4.4) and (4.7)) are necessary in order to determine these maximum values.

The differential equation that determines the motor speed $\omega_{\mathrm{m}}(t)$ reads as

$$\underbrace{\frac{\mathrm{d}}{\mathrm{d}t}\varphi_{\mathrm{m}}(t)}_{\text{speed}} = \omega_{\mathrm{m}}(t). \tag{5.7}$$

Assuming a stiff coupling of the load to the motor and neglecting friction leads to

$$\underbrace{\frac{\mathrm{d}}{\mathrm{d}t}\omega_{\mathrm{m}}(t)}_{\text{acceleration}} = \frac{K_{\mathrm{t}}}{J_{\mathrm{m}} + J_{\mathrm{l}}} i(t), \tag{5.8}$$

which represents the acceleration depending on a *single* state, namely the current $i(t)$. Hence, it is possible to consider the maximum value of the current $i_{\max}$ in the planning of position trajectories through the choice of the maximum acceleration.

By differentiation of the acceleration, the jerk reads as

$$\underbrace{\frac{\mathrm{d}^2}{\mathrm{d}t^2}\omega_{\mathrm{m}}(t)}_{\text{jerk}} = \frac{K_{\mathrm{t}}}{J_{\mathrm{m}} + J_{\mathrm{l}}} \frac{\mathrm{d}}{\mathrm{d}t} i(t), \tag{5.9}$$

where the jerk now depends on the derivative of the current $\frac{\mathrm{d}}{\mathrm{d}t} i(t)$ which means that the maximum jerk is proportional to the maximum current derivative $\frac{\mathrm{d}}{\mathrm{d}t} i(t)\Big|_{\max}$.

The current dynamics with a neglected back electromotive force (EMF),

$$\frac{\mathrm{d}}{\mathrm{d}t} i(t) = -\frac{R}{L} i(t) + \frac{1}{L} v(t), \tag{5.10}$$

show that the current derivative, and therefore the jerk, depend on the value of the current $i(t)$ and the voltage $v(t)$. Considering (5.10), the choice of the maximum jerk allows to respect the voltage limit $v_{\max}$.

Figure 5.1 shows the current response $i(t)$ to a constant input voltage $v(t) = v_{\max}$ with an initial current $i(0) = -i_{\max}$ to account for the worst-case current change from $-i_{\max}$ to $i_{\max}$. Two possibilities for the choice of the maximum current derivative $\frac{\mathrm{d}}{\mathrm{d}t} i(t)\Big|_{\max}$, which is proportional to the jerk, are shown. Approach (A) uses the slope of the tangent to the current trajectory $i(t)$ at $i_{\max}$ to determine the maximum jerk. Approach (B) determines the maximum jerk through the slope of the secant that connects the initial current $i(0) = -i_{\max}$ and maximum current $i_{\max}$.

Table 5.2 shows characteristic values for approaches (A) and (B) for a linear current trajectory starting at $-i_{\max}$ with a constant slope of $\frac{\mathrm{d}}{\mathrm{d}t} i(t)\Big|_{\max}$. For this evaluation, the linearly increasing current trajectory

$$i(t) = t \frac{\mathrm{d}}{\mathrm{d}t} i(t)\Big|_{\max} - i_{\max} \tag{5.11}$$

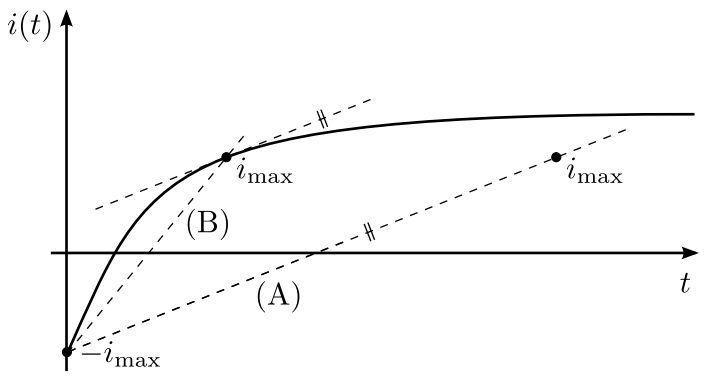


Figure 5.1: Choice of the maximum jerk illustrated with a current response to $v(t) = v_{\max}$. The initial current value is $i(0) = -i_{\max}$ to account for the worst-case current change. (A) Tangent at $i_{\max}$. (B) Secant from $-i_{\max}$ to $i_{\max}$.

is realized through the linearly increasing voltage trajectory

$$v(t) = R\left(t\frac{\mathrm{d}}{\mathrm{d}t}i(t)\bigg|_{\max} - i_{\max}\right) + L\,\frac{\mathrm{d}}{\mathrm{d}t}i(t)\bigg|_{\max}. \tag{5.12}$$

The voltage $v_{i=i_{\max}}$ is reached when the current reaches its maximum value $i = i_{\max}$ at $t_{i=i_{\max}}$. Approach (A) leads to a voltage $v_{i=i_{\max}} = 28\,\mathrm{V}$, which is exactly the maximum voltage $v_{\max} = 28\,\mathrm{V}$. By taking the tangent at $i_{\max}$, it is ensured that the maximum voltage is only reached when the maximum current is reached at $t_{i=i_{\max}} = 165.2\,\mu\mathrm{s}$. Hence, method (A) maintains the voltage constraints if the current is between $-i_{\max}$ and $i_{\max}$. However, for position trajectories that require only low current values (e.g., small step heights), this choice is conservative in terms of constraint exploitation. Approach (B), which is based on the current secant, cannot guarantee voltage constraint satisfaction. As shown in Table 5.2, a voltage of $v = 41.2\,\mathrm{V}$ would be reached when the current reaches its maximum value at $t_{i=i_{\max}} = 77.9\,\mu\mathrm{s}$. The maximum voltage $v_{\max}$ would be exceeded for a time span of $T_{v>v_{\max}} = 30.4\,\mu\mathrm{s}$, which corresponds to about three sampling instants. Nevertheless, the time when the maximum current is reached is $t_{i=i_{\max}} = 77.9\,\mu\mathrm{s}$, compared with $t_{i=i_{\max}} = 165.2\,\mu\mathrm{s}$ for approach (A). It is shown later on that for the application considered here approach (B) leads to a good trade-off between a fast response and the handling of constraints—the few sampling instants where the voltage constraints are exceeded do not lead to unsatisfactory results. Approach (A) is not used, as first tests showed slow responses.

If an application has a non-negligible back EMF, the current evolution for the *worst-case* back EMF (proportional to the worst-case speed) has to be considered—this introduces conservatism for slow speeds.

The process of the choice of the maximum jerk emphasize that it is advantageous to directly incorporate the voltage constraints in a control scheme—as it is done in the design

Approach	$\frac{\mathrm{d}}{\mathrm{d}t}i(t)\Big\vert_{\max}$ [kA s^{-1}]	$t_{i=i_{\max}}$ [µs]	$T_{v>v_{\max}}$ [µs]	$v_{i=i_{\max}}$ [V]
(A) Tangent at $i_{\max}$	60.52	165.2	0.0	28.0
(B) Secant from $-i_{\max}$ to $i_{\max}$	128.36	77.9	30.4	41.2

Table 5.2: Characteristic values concerning the choice of the maximum jerk for approaches (A) and (B). These values result from a linear current trajectory starting at $-i_{\max}$ with a slope of $\frac{\mathrm{d}}{\mathrm{d}t}i(t)\Big\vert_{\max}$.

of PRG schemes—instead of limiting the jerk, which often leads to a trade-off between a fast response and the handling of constraints. Furthermore, if voltage constraints have to be respected in any case, the limitation of the jerk can lead to conservative results.

The use of a jerk limited trajectory to account for the voltage limit is especially important when the current and position dynamics lie within the same range. Nevertheless, there are many applications where a current control loop can be considered sufficiently fast. In these applications, reasons for a constrained jerk are often the reduction of vibrations (e.g. in robotics and computer numerical control (CNC) machining) or an increased comfort (e.g. in elevator applications through a smoother reference trajectory).

5.3.3 Minimum-Time Optimal Control: The Benchmark Problem

Minimum-time optimal control (Kirk, 2004) serves as a benchmark for the two PRG approaches—it delivers the system input trajectory $\boldsymbol{u}^\star(t)$ which steers the system from one state to another in minimum-time while respecting system constraints. Hence, this benchmark problem delivers the minimum final time t_{final} that is *physically* possible. In contrast to PRG based approaches, there is *no* reference *trajectory* in minimum-time optimal control, the reference consists of an initial state $\boldsymbol{x}_{\mathrm{initial}}$ and a final state $\boldsymbol{x}_{\mathrm{final}}$ which should be reached in minimum time. This fact represents a difference between the PRG approach and minimum-time optimal control—PRGs can deal with *arbitrary* varying reference signals. Nonetheless, this work shows that it is possible to approximate minimum-time optimal control with an ℓ_1-norm based PRG. Solving a minimum-time optimal control problem is, in general, computationally more complex than solving the optimization problems that arise for PRGs. For simple systems, like a double integrator, it is possible to derive an analytic solution to the minimum-time optimal control problem. More complicated problems, however, often lead to nonlinear programs (Rao et al., 2010), which generally take longer to be solved than the optimization problems of PRGs.

A minimum-time optimal control problem is represented by

$$\begin{aligned} \underset{\boldsymbol{u}(t),\boldsymbol{x}(t)}{\text{minimize}} \quad & t_{\text{final}} \\ \text{subject to} \quad & (4.4) \\ & \boldsymbol{x}(0) = \boldsymbol{x}_{\text{initial}}, \\ & \boldsymbol{x}(t_{\text{final}}) = \boldsymbol{x}_{\text{final}}, \\ & |\boldsymbol{u}(t)| \leq \boldsymbol{u}_{\max}, \\ & |\boldsymbol{x}(t)| \leq \boldsymbol{x}_{\max}. \end{aligned} \tag{5.13}$$

The aim is to minimize the final time t_{final} considering the continuous system dynamics (4.4), while maintaining input and state constraints. The boundary conditions are the initial state $\boldsymbol{x}_{\text{initial}}$ and the final state $\boldsymbol{x}_{\text{final}}$. As the system considered for the experimental validation is a SISO system, the number of inputs is $n_u = 1$. The number of outputs n_y is not relevant here, as the complete state vectors $\boldsymbol{x}_{\text{initial}}$ and $\boldsymbol{x}_{\text{final}}$ are needed for this sort of optimal control problem.

A solution for the minimum-time optimal control problem (5.13) exists if the real parts of the eigenvalues of $\boldsymbol{A}_c$ (see (4.4)) are non-positive (Pontryagin et al., 1962). It is well known that, if it exists, the optimal input trajectory $\boldsymbol{u}^\star(t)$ for linear time-invariant (LTI) systems with input constraints is bang-bang (Kirk, 2004, p. 249), which means that it only consists of extreme realizations $\boldsymbol{u}^\star(t) \in \{-\boldsymbol{u}_{\max}, \boldsymbol{u}_{\max}\}$. Furthermore, if this solution exists, it is unique (Pontryagin et al., 1962). If all n_x eigenvalues of $\boldsymbol{A}_c$ are real and non-positive, the bang-bang input trajectory consists of *at most* $n_{\text{sw}} = (n_x - 1)$ switchings (Kirk, 2004, p. 249). The transition from $\boldsymbol{u}_{\max}$ to $-\boldsymbol{u}_{\max}$, or vice versa, is called switching. For complex eigenvalues, the number of switchings n_{sw} is limited but no analytical value for an upper bound can be given.

For the plant that is treated here, all eigenvalues of the system matrix $\boldsymbol{A}_c$ have a non-positive real part. Hence, a unique (bang-bang) solution to the minimum-time optimal control problem exists. The system matrix $\boldsymbol{A}_c$ has three real and two complex eigenvalues originating from the two-mass behavior. Therefore, the number of switchings n_{sw} is limited, but the limit is unknown.

The optimal control problem (5.13) also includes constraints on the states. If these state constraints get active, the resulting input trajectory does not contain extreme realizations only.

The reason for choosing a *continuous-time* optimal control problem is that the principal shape of the resulting trajectories is known *before* the problem is solved. This makes it easy to validate the results of a respective optimal control problem solver. *Discrete-time* minimum-

time optimal control problems, which directly take into account the discrete nature of the control system, are treated, for example, in (Rothwangl, 2001; Van den Broeck et al., 2009; Chen et al., 2012). First tests comparing these three approaches showed that they deliver the same discrete final time. The respective trajectories were, however, not the same—this can be explained by the fact that the continuous minimum final time might lie between two sampling instants. The discrete-time approaches deliver the next higher sampling instant (compared with the continuous-time solution). Since this time is not exactly minimal anymore, there exist multiple trajectories that achieve this result. This observation is a further reason why the continuous-time optimal control problem is solved in this work.

The minimum-time optimal control problem (5.13) is solved using the optimal control toolbox GPOPS 4.1 (Rao et al., 2010) for MATLAB, which transforms continuous-time optimal control problems to nonlinear programming problems. This toolbox delivers optimal continuous-time[2] trajectories for the input $\boldsymbol{u}^\star(t)$ and the state $\boldsymbol{x}^\star(t)$.

A remaining issue is the interaction between the continuous-time control problem (5.13) and the discrete-time control system on which it is implemented. Traditionally, it is assumed that the sampling time T_s of the control system is sufficiently small compared to the plant dynamics. The sampling time $T_\mathrm{s} = 10\,\mu\mathrm{s}$ of the application considered here is relatively large compared to the plant dynamics. The experimental validation is based on the dynamic, model-based feedforward control (DynFF) concept of Section 4.2.2.2, as the trajectories of the input $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ and the state $\boldsymbol{x}^{\mathrm{r,opt}}[k]$ are available.

The optimal state trajectory $\boldsymbol{x}^\star(t)$ is discretized using the zero-order hold (ZOH) method

$$\boldsymbol{x}^{\mathrm{r,opt}}[k] = \boldsymbol{x}^\star(kT_\mathrm{s}). \tag{5.14}$$

However, it turned out that the ZOH method applied to the optimal input trajectory $\boldsymbol{u}^\star(t)$ leads to non-satisfactory results. The input $\boldsymbol{u}^\star(t)$ consists, in the extreme case, only of switchings between extreme realizations. In general, the switching times do *not* coincide with the sampling instants of the discrete-time control system. Hence, to reproduce this input shape, a very fine time resolution of the discrete-time system would be necessary. However, as the involved sampling time $T_\mathrm{s} = 10\,\mu\mathrm{s}$ cannot be lowered further, it is proposed to discretize the plant optimal input trajectory $\boldsymbol{u}^\star(t)$ as

$$\boldsymbol{u}^{\mathrm{r,opt}}[k] = \frac{1}{T_\mathrm{s}} \int_{kT_\mathrm{s}}^{(k+1)T_\mathrm{s}} \boldsymbol{u}^\star(t)\,\mathrm{d}t. \tag{5.15}$$

[2] *Continuous-time* means in this case that the signals are available with a much higher time-resolution than the sampling time T_s of the discrete-time control system.

This approach preserves the input-time-area when the continuous-time signal is discretized. The validity of this discretization method is shown experimentally later on.

5.3.4 Experimental Results

The experimental results for all four discussed approaches of this chapter are presented here. This extensive comparison allows to classify the PRG approaches in between the advanced industry standard and the minimum-time solution, which represents the physical limit. The experimental results show step responses for different step heights to evaluate the different responses concerning the performance criteria and the handling of constraints. The reason for choosing a reference in the shape of a step is that the minimum-time solution can be gathered relatively easily (compared to other reference shapes) and that there are common performance indicators for step responses that allow a simple comparison of the approaches.

The digital control system introduces a delay of one sampling instant because of the time it takes to calculate the control algorithm. This so-called computational delay was compensated in all approaches of this thesis through a shift of the reference signal by one step.

The preview feature of the two PRGs of this chapter was disabled in order to allow a fair comparison to the other two approaches (advanced industry standard and minimum-time optimal control). All important PRG parameters are shown in Table 5.1.

The experimental results shown here involve a SISO system and therefore a single motor[3] (axis 2) of the positioning system of Chapter 4.

Performance criteria

The following common performance criteria for a step response are used to compare the four discussed approaches:

- Overshoot (OS): Peak value of the step response in [%] of the reference step height
- Rise time (RT): Time in sampling instants $[kT_s]$ to bring the system output from 10 % to 90 % of the reference step height
- Settling time (ST): Time in sampling instants $[kT_s]$ to bring the system output into the common error band of ± 2 % of the reference step height

Dealing with minimum-time optimal control, an interesting value is the number of switchings n_{sw} that the bang-bang profile exhibits. The number of switchings depends on the initial and final states, as complex eigenvalues are involved here.

[3] The results of axis 1 are omitted because they show a similar behavior. Results for a simultaneous application of *both* axes (biaxial contouring) are shown in Chapter 7.

Another important performance measure is the control error $e^{\mathrm{r}}[k]$

$$e^{\mathrm{r}}[k] = \varphi_{\mathrm{l}}^{\mathrm{r}}[k] - \varphi_{\mathrm{l}}[k], \tag{5.16}$$

which gives a measure of the tracking accuracy of the plant output $\varphi_{\mathrm{l}}[k]$ compared with the *unoptimized* reference $\varphi_{\mathrm{l}}^{\mathrm{r}}[k]$.

A further performance measure is the error between the optimized reference $\varphi_{\mathrm{l}}^{\mathrm{r,opt}}[k]$ and the plant output $\varphi_{\mathrm{l}}[k]$

$$e^{\mathrm{r,opt}}[k] = \varphi_{\mathrm{l}}^{\mathrm{r,opt}}[k] - \varphi_{\mathrm{l}}[k]. \tag{5.17}$$

This here called optimized control error $e^{\mathrm{r,opt}}[k]$ is ideally zero, which corresponds to perfect tracking of the optimized reference. The optimized control error is deteriorated by model mismatch and if hard constraints are not respected.

Description of an exemplary experimental results figure

As the four approaches of this chapter are validated for different step heights, this paragraph introduces the notation and colors of the presented figures in order to keep the repeating figure captions simple and focused.

Figure 5.3b on page 89 serves as an example here:

- First plot (from top): Reference load position $\varphi_{\mathrm{l}}^{\mathrm{r}}[k]$ (——), load position $\varphi_{\mathrm{l}}[k]$ (——), motor position $\varphi_{\mathrm{m}}[k]$ (——). The dash dotted lines (-·-·-) show the 10 % and 90 % values of the reference to determine the RT. The optimized reference $\varphi_{\mathrm{l}}^{\mathrm{r,opt}}[k]$ is *not* shown, as this value almost overlaps with $\varphi_{\mathrm{l}}[k]$. Optional continuous-time signals (minimum-time solution) are $\varphi_{\mathrm{l}}(t)$ (- - - -) and $\varphi_{\mathrm{m}}(t)$ (- - - -).

- Second plot: This is a zoomed version of the first plot to determine the ST. The ±2 % lines (- - - -) show the error band. Furthermore, a possible overshoot is clearly visible in this plot.

- Third plot: Control error $e^{\mathrm{r}}[k]$ (——)

- Fourth plot: Optimized control error $e^{\mathrm{r,opt}}[k]$ (——). Even though $\varphi_{\mathrm{l}}^{\mathrm{r,opt}}[k]$ is not shown in the first two plots, the optimized control error $e^{\mathrm{r,opt}}[k]$ allows to evaluate the tracking of the optimized reference.

- Fifth plot: Load speed $\omega_{\mathrm{l}}[k]$ (——), motor speed $\omega_{\mathrm{m}}[k]$ (——). Optional continuous-time signals (minimum-time solution) are $\omega_{\mathrm{l}}(t)$ (- - - -) and $\omega_{\mathrm{m}}(t)$ (- - - -).

- Sixth plot: Motor current $i[k]$ (——) and current constraints $\pm i_{\max}$ (- - - -). An optional continuous-time signal (minimum-time solution) is $i(t)$ (- - - -).
- Bottom plot: Motor voltage $v[k]$ (——) and voltage constraints $\pm v_{\max}$ (- - - -). The complete range of the ordinate represents the hard input voltage constraints $\pm v_{\mathrm{DC}}$. An optional continuous-time signal (minimum-time solution) is $v(t)$ (- - - -).

In general, the color scheme uses red for load-side values ($\varphi_{\mathrm{l}}[k]$ and $\omega_{\mathrm{l}}[k]$) and values that depend on the load position ($e^{\mathrm{r}}[k]$, $e^{\mathrm{r,opt}}[k]$, etc.). Blue is used for the motor-side values ($v[k]$, $i[k]$, $\varphi_{\mathrm{m}}[k]$, and $\omega_{\mathrm{m}}[k]$). Constraints ($v_{\max}$ and $i_{\max}$) are indicated by black dashed lines (- - - -). These general conventions are valid for the whole thesis unless stated otherwise.

Discussion of the experimental results

The experimental results for a step reference with a height of $\varphi_{\mathrm{l}}^{\mathrm{r}} = 0.5\,\mathrm{mrad}$ are shown in Figures 5.2 and 5.3 on pages 88/89 to allow a convenient comparison. The reference step is applied at $t = 0.4\,\mathrm{ms}$ for all four approaches, which results in a control action (voltage v) at $t = 0.4\,\mathrm{ms}$, as the computational delay was compensated. The control action affects the states at $t = 0.41\,\mathrm{ms}$ when the next measurements takes place.

The results for the advanced industry standard (acceleration- and jerk-limited trajectory) are shown in Figure 5.2a. The reference value of $\varphi_{\mathrm{l}}^{\mathrm{r}} = 0.5\,\mathrm{mrad}$ is reached without an overshoot. The voltage constraints $\pm v_{\max}$, however, are *not* fully exploited; this results from the relatively low current values that are reached (see approach (B) in Section 5.3.2). The fact that these constraints are not exploited indicates that the reference value is not reached in minimum-time. The values of the RT and the ST are therefore comparably high (numerical values are shown in Table 5.3 on page 94). The optimized control error $e^{\mathrm{r,opt}}[k]$ is comparably high because of a delay of two sampling instants, which is introduced by designing a *causal* zero-phase-error tracking control (ZPETC) feedforward transfer function (TF). The plot of the optimized control error $e^{\mathrm{r,opt}}[k]$ is cropped for the advanced industry standard to allow the evaluation of the small values that the other three approaches show.

Figure 5.2b shows the experimental results of the standard PRG approach. The voltage constraints are well exploited which leads to a small RT. This small RT, however, comes with a relatively high OS. Due to this OS, the ST of the standard PRG is in the same range as the ST of the advanced industry standard. The optimized control error $e^{\mathrm{r,opt}}[k]$ is in the range of $\pm 10\,\mu\mathrm{rad}$ which is just slightly above the noise level. This small error indicates nearly perfect tracking of the optimized reference through the DynFF concept. Hence, the overshoot is a result of the optimization that takes place in the PRG and is *not* a result of, for example, a model mismatch. A small optimized control error $e^{\mathrm{r,opt}}[k]$ is only achievable through a

precise system model and the handling of constraints—especially the voltage constraints of $\pm v_{\max}$ need to be respected through the optimized reference. The voltage constraint are of special importance because the control reserve (CR) is relatively small and therefore a violation of $\pm v_{\max}$ can easily lead to a violation of the *hard* voltage constraints of $\pm v_{\mathrm{DC}}$.

The resulting signals for the near minimum-time PRG are presented in Figure 5.3a. The uniform weighting of the control error by the use of the ℓ_1-norm leads to a reduced overshoot, which is even in the error band of the ST. This leads to a reduced ST compared with the STs of the two previously discussed approaches. Again, the optimized control error $e^{\mathrm{r,opt}}[k]$ is in the range of $\pm 10\,\mu\mathrm{rad}$. The shape of the input voltage $v[k]$ almost entirely consists of extreme realizations of $\pm v_{\max}$. The behavior of the two-mass system is clearly visible—the motor-side values ($\varphi_{\mathrm{m}}[k]$ and $\omega_{\mathrm{m}}[k]$) differ significantly from the respective load-side values ($\varphi_{\mathrm{l}}[k]$ and $\omega_{\mathrm{l}}[k]$). Especially at $t = 0.45\,\mathrm{ms}$, the load speed $\omega_{\mathrm{l}}[k]$ reaches its maximum value, whereas the motor speed $\omega_{\mathrm{m}}[k]$ is nearly zero. Idle control, which is a disadvantage of ℓ_1-norm based cost functions, was not observed in the experiments that were carried out for various step heights. Deadbeat behavior, which is considered to be a disadvantage in literature (Rao and Rawlings, 2000), is exploited here to achieve a near minimum-time transient response.

The minimum-time optimal control problem, which serves as a benchmark in this work, is shown in Figure 5.3b. The reference step is also plotted in this figure for timing and comparison reasons although there is no real reference. The reference only consists of the initial state $\boldsymbol{x}_{\mathrm{initial}} = \left[\,0\,\mathrm{A}\;\; 0\,\mathrm{mrad}\;\; 0\,\mathrm{rad\,s^{-1}}\;\; 0\,\mathrm{mrad}\;\; 0\,\mathrm{rad\,s^{-1}}\,\right]^\top$ and the final state $\boldsymbol{x}_{\mathrm{final}} = \left[\,0\,\mathrm{A}\;\; 0.5\,\mathrm{mrad}\;\; 0\,\mathrm{rad\,s^{-1}}\;\; 0.5\,\mathrm{mrad}\;\; 0\,\mathrm{rad\,s^{-1}}\,\right]^\top$. The continuous-time signals, resulting from the minimum-time optimal control solver, are depicted as dashed lines. The good agreement of the optimized continuous-time state signals with the experimental discrete-time signals validates the proposed discretization method (5.15). The discrete-time input voltage is—compared with the continuous one—not bang-bang anymore, but it preserves the voltage-time area. The continuous-time bang-bang voltage profile shows $n_{\mathrm{sw}} = 4$ switchings. An OS is barely visible. The RT and the ST are equal to the values of the near minimum-time PRG.

The respective signals for the four approaches for a step height of $\varphi_{\mathrm{l}}^{\mathrm{r}} = 2\,\mathrm{mrad}$ are shown in Figures 5.4 and 5.5 on pages 90/91. The input voltage $v[k]$ of the advanced industry standard approach is again conservative but already higher than the input voltage of the 0.5 mrad step. However, this response is again obviously not time-optimal. For this step height, the current $i[k]$ of the two PRGs slightly touches the current constraints of $\pm i_{\max}$. Furthermore, the minimum-time solution now shows six switchings $n_{\mathrm{sw}} = 6$, which confirms that this number depends on the initial and final values for the two-mass system of this work

(complex eigenvalues). The minimum-time current trajectory $i[k]$ does not yet saturate—this is the reason why the input $v[k]$ again only consists of extreme realizations. The near minimum-time PRG again shows a much smaller OS compared with the standard PRG. Nevertheless, the OS of the near minimum-time approach is slightly outside the ST error band; this leads to ST that is slightly higher than the minimum-time ST.

The four approaches for a step height of $\varphi_\mathrm{l}^\mathrm{r} = 10\,\mathrm{mrad}$ are presented in Figures 5.6 and 5.7 on pages 92/93. In contrast to the smaller step heights, the voltage limit of $\pm v_\mathrm{max}$ is now even exceeded for time spans of about three sampling instants (see approach (B) in Section 5.3.2). The shape of the current trajectory shows that a constant (limited) jerk leads to a nearly constant current slope. The current limitation is reached but not exceeded by both PRGs. The ST of the near minimum-time approach is equal to the ST of the minimum-time solution. The minimum-time current response also exploits the full current range. As input *and* state constraints are active now, the continuous-time input profile is *not* bang-bang anymore. Comparing the continuous-time current trajectory $i(t)$ (solution of the minimum-time problem) with the discrete-time voltage $i[k]$ (determined experimentally) emphasizes—together with a small error $e^{\mathrm{r,opt}}[k]$—the quality of the plant model. The good agreement of the continuous- and discrete-time currents even for currents in the full range of $\pm i_\mathrm{max}$ validates—besides the quality of the proposed discretization method—that it is a reasonable approach to model the electrical subsystem as an LTI system, i.e., neglecting magnetic saturation.

Table 5.3 on page 94 compares the performance criteria of the four discussed methods for different reference step heights. The advanced industry standard, in general, does not show an OS. Nevertheless, the STs of the advanced industry standard are considerably higher than the minimum-time STs—especially for low step heights. The weights of the PRG formulations are kept constant, they are tuned for the step height of $\varphi_\mathrm{l}^\mathrm{r} = 0.5\,\mathrm{mrad}$ (marked in gray in Table 5.3). Throughout this table, the near minimum-time PRG shows an ST that is close to the ST of the minimum-time approach, but shows a small OS that is usually inside the ST error band. For the step height of $\varphi_\mathrm{l}^\mathrm{r} = 5\,\mathrm{mrad}$, the ST of the near minimum-time PRG is, due to a small overshoot, even one sampling instant smaller than the ST of the minimum-time optimal control problem. The step response of minimum-time optimal control does not show an OS. The standard PRG approach shows compared with the near minimum-time PRG a higher OS and a significantly higher ST especially for small step heights. Table 5.3 also illustrates that the near minimum-time approach, compared with the standard PRG, shows a smaller variation in the performance criteria for different reference step heights. This suggests that, at least for the application treated here, the ℓ_1-norm based cost function leads to a quite balanced near minimum-time performance for a broad range of

reference values. The prediction horizon $N = 20$ does not cover the entire step dynamics for $\varphi_\mathrm{l}^\mathrm{r} = 5\,\mathrm{mrad}$ and $\varphi_\mathrm{l}^\mathrm{r} = 10\,\mathrm{mrad}$—nevertheless, the results are still satisfying. The reference range up to $\varphi_\mathrm{l}^\mathrm{r} = 10\,\mathrm{mrad}$ is quite realistic for the positioning system of this work. However, in order to achieve comparable control results also for step heights larger than 10 mrad, it would be necessary to increase the prediction horizon N to cover to the whole step dynamics. Another interesting observation is that the number of switchings n_sw for the time-optimal problem varies depending on the step height (here $n_\mathrm{sw} = 4$ or $n_\mathrm{sw} = 6$). The occurrence of $n_\mathrm{sw} = 6$ confirms that n_sw is not bounded by $(n_x - 1) = 4$ because complex eigenvalues are involved.

Summary

Summing up, the experimental results show that an ℓ_1-norm based PRG (near minimum-time formulation) is able to approximate minimum-time optimal control quite well, while representing a computationally less complex optimization problem. Furthermore, it is shown that a near minimum-time behavior is achieved for a broad range of reference values without the need of re-tuning. The reference signals in the shape of a step allow to directly compare the two presented PRG approaches with the minimum-time solution of an optimal control problem. The solution of this problem exhibits the minimum final time that is *physically* possible. The presented PRGs—unlike the minimum-time optimal control problem—can deal with arbitrary varying reference signals.

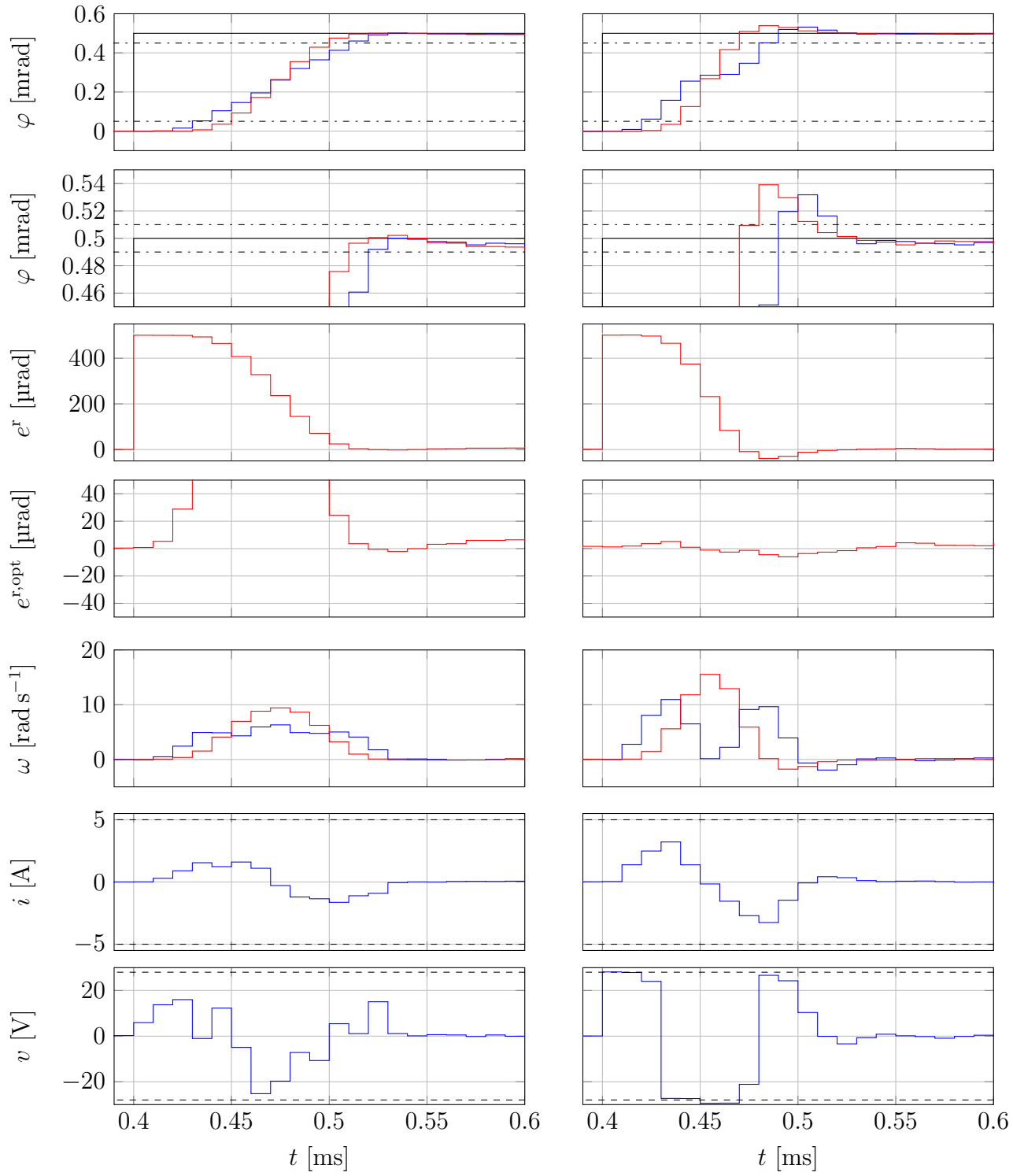
5.3.5 Computation Time

The used software to solve the optimization problems arising in this work is shortly discussed in Section 2.1.6. Optimization of the computation time of the respective algorithms is not the focus of this work, much more the principle and the applicability of the schemes are important. Nevertheless, this section gives a rough estimation of the achievable sampling times using the two PRG schemes of this chapter[4].

The investigations concerning the reachable sampling time were carried out with an Intel® Core™ i5 @2.5GHz processor (only one core was used) by using MATLAB for pre- and post-calculations and a C++ implementation of the solver qpOASES (Ferreau et al., 2008) for solving the optimization problem. The minimum sampling time consists of pre- and post-calculations and the *maximum* solver time of the respective optimization algorithm, which is usually reached when constraints get active. The pre- and post-calculations were estimated to be 0.5 ms. The standard PRG approach led to a maximum solver time of 3.8 ms, which makes a sampling time of 4.3 ms possible. The problem size of the near minimum-time PRG optimization problem is larger than the one of a standard PRG, because of the necessary transformation to an LP. Hence, only a maximum solver time of 17 ms was achieved, leading to a sampling time of 17.5 ms. These investigations rely on the parameters of Table 5.1 on page 75 including state and input constraints. The given estimated sampling times are based on a receding horizon shift of $N_{\text{shift}} = 1$. For example, assuming an offline operation of a PRG, choosing $N_{\text{shift}} = 5$ would lead to a decreased overall computation time by factor 5. Assuming an online operation of a PRG, $N_{\text{shift}} = n_{\text{mr}} = 5$ would allow a sampling time of $5 \cdot T_{\text{s}}$ for the PRG, which allows a larger optimization problem size (e.g., a larger prediction horizon or more constraints).

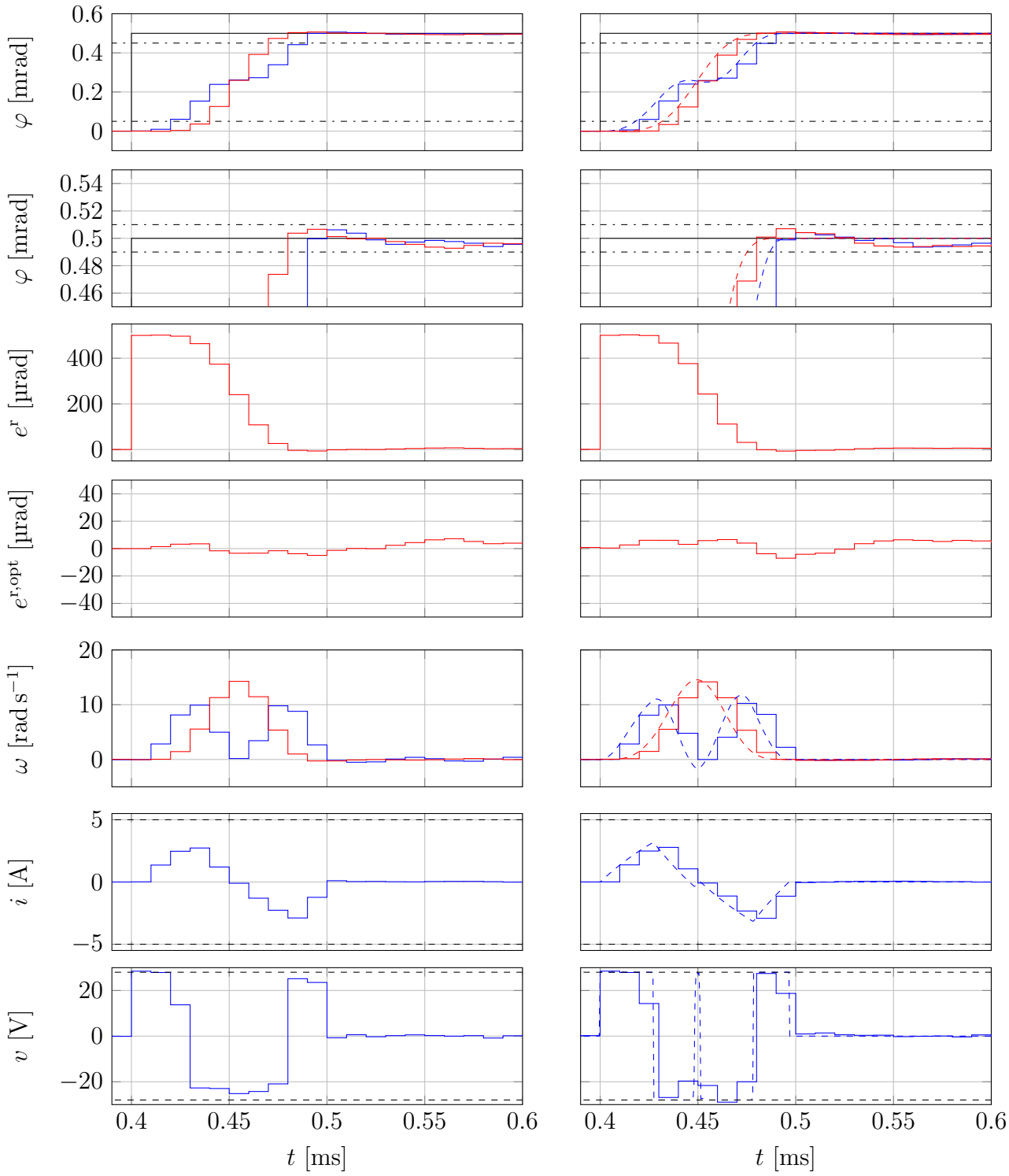
Summing up, the reachable sampling time is, up to now, not suitable for the application considered here, which needs a maximum sampling time of $T_{\text{s}} = 10\,\mu\text{s}$. Even though the real-time execution of the PRG schemes is not the focus of this thesis, the steadily increasing computational power of embedded hardware suggest that the presented PRG algorithms will soon be applicable for real-time operation even on fast sampling systems. Furthermore, the steady improvement of optimization algorithms, especially for parallel computing on field programmable gate arrays (FPGAs) (Jerez et al., 2013), is the driving force behind real-time PRG applications.

[4]The computation times of the approaches of the following chapters are not given because this section already gives a first impression of the involved computation times.



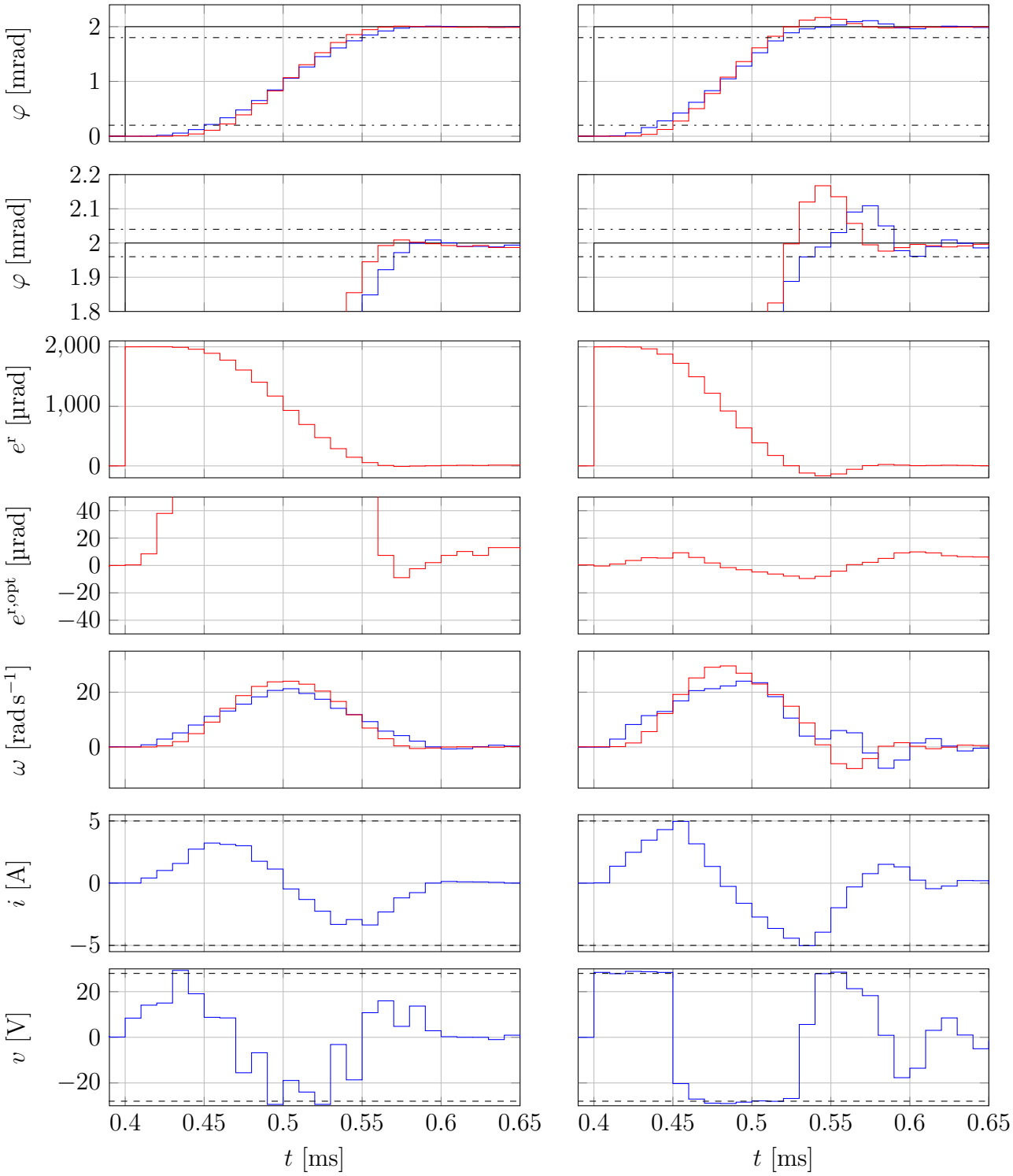
(a) Advanced industry standard (b) Standard PRG

Figure 5.2: Step response comparison ($\varphi_1^{\mathrm{r}} = 0.5\,\mathrm{mrad}$) for the advanced industry standard and the standard PRG (see page 82 for the description of the signals)



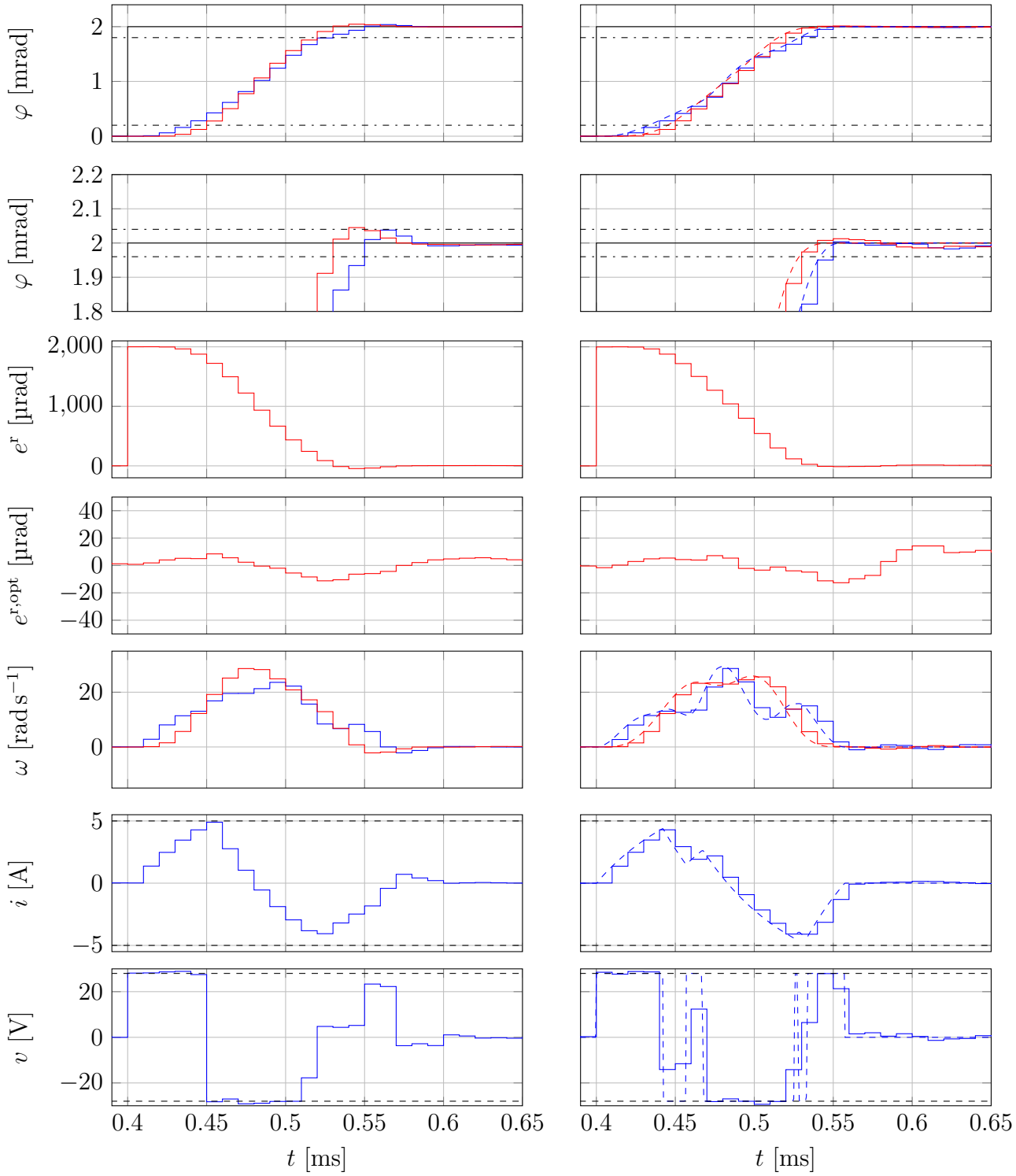
(a) Near minimum-time PRG (b) Minimum-time

Figure 5.3: Step response comparison ($\varphi_1^r = 0.5\,\text{mrad}$) for the near minimum-time PRG and the minimum-time optimal control problem (see page 82 for the description of the signals)



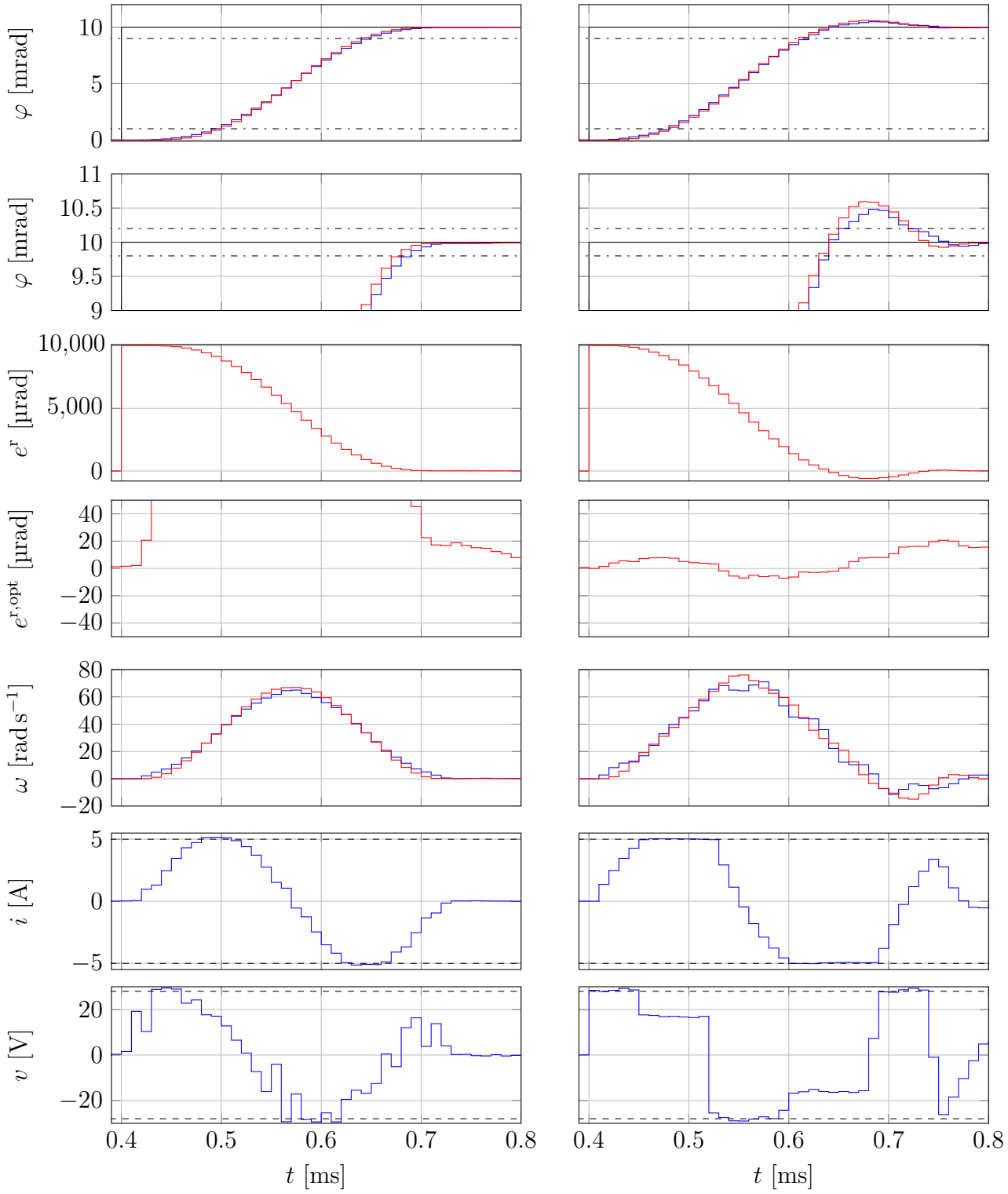
(a) Advanced industry standard (b) Standard PRG

Figure 5.4: Step response comparison ($\varphi_1^{\mathrm{r}} = 2\,\mathrm{mrad}$) for the advanced industry standard and the standard PRG (see page 82 for the description of the signals)



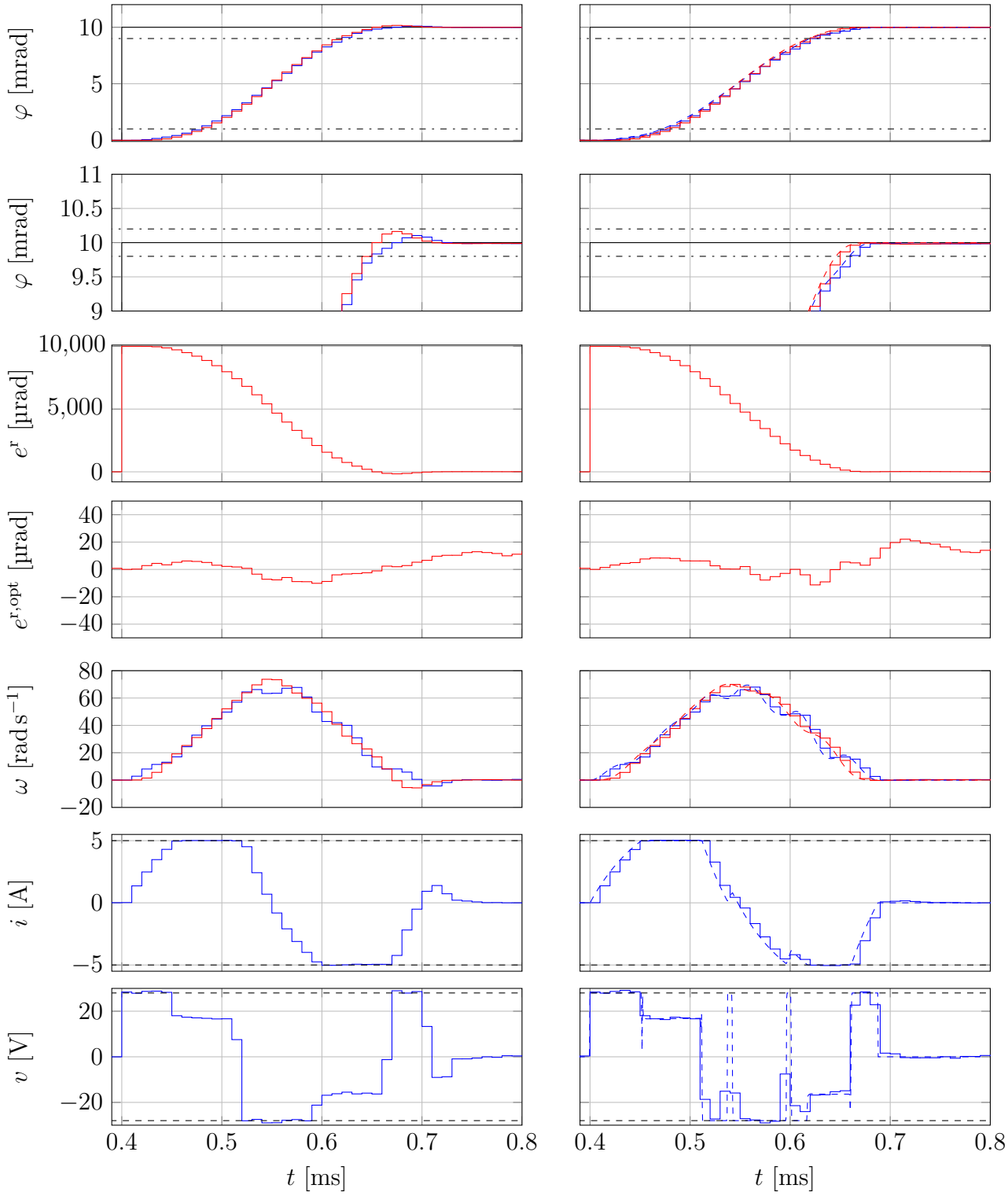
(a) Near minimum-time PRG (b) Minimum-time

Figure 5.5: Step response comparison ($\varphi_1^r = 2\,\text{mrad}$) for the near minimum-time PRG and the minimum-time optimal control problem (see page 82 for the description of the signals)



(a) Advanced industry standard (b) Standard PRG

Figure 5.6: Step response comparison ($\varphi_1^{\mathrm{r}} = 10\,\mathrm{mrad}$) for the advanced industry standard and the standard PRG (see page 82 for the description of the signals)



(a) Near minimum-time PRG (b) Minimum-time

Figure 5.7: Step response comparison ($\varphi_1^{\mathrm{r}} = 10\,\mathrm{mrad}$) for the near minimum-time PRG and the minimum-time optimal control problem (see page 82 for the description of the signals)

Step height φ_1^{r} [mrad]	Advanced industry standard			Standard PRG			Near minimum-time PRG			Minimum-time			
	OS [%]	RT [kT_s]	ST [kT_s]	OS [%]	RT [kT_s]	ST [kT_s]	OS [%]	RT [kT_s]	ST [kT_s]	OS [%]	RT [kT_s]	ST [kT_s]	n_sw
0.1	-	5	9	8.5	3	15	1.6	4	7	-	4	7	4
0.2	-	5	10	12.5	3	16	0.3	4	8	-	4	8	4
0.5	-	6	12	7.8	4	12	1.3	4	9	-	4	9	4
1.0	-	7	14	9.4	6	15	3.9	6	15	-	7	12	6
2.0	-	9	17	8.3	7	18	2.3	8	16	-	8	14	6
5.0	-	12	22	6.5	11	25	1.1	11	19	-	12	20	-
10.0	-	15	29	5.9	14	33	1.6	15	26	-	15	26	-

Table 5.3: Performance comparison for the four discussed approaches for different step heights. The grey row emphasizes the nominal step height, which is $\varphi_1^{\mathrm{r}} = 0.5\,\mathrm{mrad}$, and that the tuning for the PRG approaches was done for this step height. The advanced industry standard and the minimum-time approach do not show an overshoot (disregarding noise). The number of switchings of the minimum-time input is given as long as it only consists of extreme realizations (current constraints not active).

6 Robust Near Minimum-Time PRGs for Constrained SISO LTI Systems

Apart from stability, robustness against uncertain plant parameters is another important property of a control system. The focus of this chapter lies on *robust performance*, which means that the dynamic performance of the overall system should be influenced as little as possible by uncertain plant parameters. Nevertheless, a controller that leads to a stable closed-loop system—even in the presence of uncertain parameters—is a prerequisite for the design of predictive reference governors (PRGs) that focus on robust performance. This chapter deals with PRGs that are based on robust model predictive control (RMPC), more precisely, on min-max RMPC based on an enumerative scheme (see Section 2.3). It is proposed in this work to use the ideas of RMPC to design PRGs that focus on robust performance.

As in Chapter 5, two PRGs are introduced here: The robust PRG is based on the RMPC approach of Section 2.3 and therefore consists of a cost function based on a quadratic form. The robust near minimum-time PRG incorporates an ℓ_1-norm based cost function—it is shown that this formulation again leads to near minimum-time results.

A single axis of the positioning system considered in this work is used to validate and compare the robust PRGs to the non-robust PRG approaches of Chapter 5. A difference to the experimental results of Chapter 5 is that now the preview feature of the PRG approaches is used to track the reference signal. The experimental evaluation of this chapter is based on a reference in the shape of a step and a ramp in order to show that PRGs can deal with arbitrary varying reference signals. The experimental results show that the robust performance can indeed be improved for the majority of the uncertain parameters considered here.

The PRG approaches again have to be executed offline ($\text{online}_{\text{PRG}} = \text{false}$) and therefore no feedback information can be used ($\text{feedback}_{\text{PRG}} = \text{false}$) due to the fast sampling control system. The lack of feedback in the PRG approaches is a further motivation to evaluate and to improve the robustness against uncertain parameters.

6.1 Robust PRG Formulation

This section introduces a robust PRG that is based on the general PRG approach of Chapter 3. The basis of this robust PRG formulation is the RMPC approach of Section 2.3.

The cost function (2.48) of the RMPC formulation is repeated here for reasons of completeness

$$J_{\mathrm{q,rob}}^{(l)}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}} \, \boldsymbol{E}^{\mathrm{r},(l)}[k] \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}} \, \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2, \tag{6.1}$$

where $l \in \{1, 2, \ldots, n_{\mathrm{seq}}\}$. The index l identifies the respective prediction sequence (see prediction tree in Figure 2.8 on page 32). The min-max based optimization problem (2.50) is also repeated

$$\begin{aligned} &\underset{\boldsymbol{U}[k]}{\text{minimize}} \quad \max_{l \in \{1,2,\ldots,n_{\mathrm{seq}}\}} J_{\mathrm{q,rob}}^{(l)}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \\ &\text{subject to} \quad \mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{6.2}$$

More details and the transformation to a second-order cone program (SOCP) are given in Section 2.3.

The robust PRG formulation is considered to be the robust counterpart of the standard PRG presented in Section 5.1. It is shown later on that the control performances of the robust PRG and the non-robust PRG are almost identical for the *nominal* case. This behavior is achieved by the use of the ideas of RMPC based on enumerative schemes (Casavola et al., 2000a; Schuurmans and Rossiter, 2000). An RMPC approach based on semidefinite relaxations (Löfberg, 2003) worked in principle, but first simulations showed a quite different (conservative) behavior from the non-robust case. As it is a main goal of this work to achieve a fast dynamic behavior, the approach based on semidefinite relaxations is not eligible. An advantage of the enumerative schemes used here is that the weighting matrices can remain the same as in the non-robust PRG formulation.

The basic properties of a quadratic form based cost function are the same for the robust PRG. The non-uniform weighting of the control error again leads to the tendency to show an overshoot in the step response. Hence, as already shown in the experimental results of Chapter 5, the reference value is not reached in minimum-time.

Compared to the standard PRG optimization problem that can be cast as a quadratic program (QP), the underlying optimization problem of a robust PRG can be cast as an SOCP. An SOCP also represents an optimization problem in standard form; a respective solver is, however, computationally more complex than a solver for a QP.

If the robust PRG is operated without feedback, it is not clear which system model should be used to simulate the plant to get the next state. As only extreme realizations of the model are known, it is reasonable to use the nominal plant model to simulate the plant. This approach represents a difference compared to a robust PRG with feedback, where the state is measured/estimated at each sampling instant. Thus, the measured/estimated state of a robust PRG with feedback originates from the uncertain plant and not from the nominal plant. Therefore, it is not clear whether the theory of RMPC, which is executed online with feedback, also holds for a robust PRG that is operated without feedback.

Existing robust PRG schemes rely on cost functions based on quadratic forms. The PRG approaches of Bemporad and Mosca (1998), Casavola et al. (2000b), and Sugie and Fukui (2003)[1] mainly focus on robust constraint fulfillment, which leads to conservative control results compared to the PRGs of this thesis.

Robust constraint fulfillment is not considered here in order to reduce the rather high computational complexity and to avoid conservatism that is often introduced by robust constraint fulfillment. Hence, it cannot be guaranteed that the constraints are fulfilled for all uncertain system realizations. The PRG outputs $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ and $\boldsymbol{x}^{\mathrm{r,opt}}[k]$, which serve as feedforward signals, respect the imposed input and state constraints. In the case of uncertain parameters the feedback controller gets active[2] and contributes together with the feedforward input $\boldsymbol{u}^{\mathrm{r,opt}}[k]$ to the total plant input $\boldsymbol{u}[k]$. Hence, it cannot be guaranteed to fulfill the input constraints, which makes it necessary to establish a control reserve (CR) to allow the feedback controller to react in the presence of uncertain parameters. The state constraints can also be slightly violated—it is shown exemplary for the current constraints that these slight violations are not critical. For applications where state constraints have to be respected for all uncertainty realizations, it is recommended to impose a state constraint reserve in the PRG formulation.

6.2 Robust Near Minimum-Time PRG Formulation

The robust near minimum-time PRG relies—like the corresponding non-robust formulation—on an ℓ_1-norm based cost function. The uniform weighting of the ℓ_1-norm is exploited to achieve near minimum-time control results.

[1] The cited literature also uses the terms *reference management*, *reference shaping*, *command governor*, *command shaping*, and *receding horizon based trajectory planning* for PRGs. Nevertheless, these approaches rely on prediction and the receding horizon approach. Thus, only the term PRG is used in this thesis.

[2] This statement is general, i.e., it is valid for robust PRGs and non-robust PRGs in combination with the dynamic, model-based feedforward control (DynFF) approach.

The cost function based on the ℓ_1-norm is given as

$$J^{(l)}_{\ell_1,\mathrm{rob}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} \boldsymbol{Q}_{\ell_1} \boldsymbol{E}^{\mathrm{r},(l)}[k] \\ \boldsymbol{R}_{\ell_1} \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_1 . \tag{6.3}$$

The predicted control error $\boldsymbol{E}^{\mathrm{r},(l)}[k]$ is weighted with the diagonal matrix $\boldsymbol{Q}_{\ell_1}$ and is dependent on $l \in \{1, 2, \ldots, n_{\mathrm{seq}}\}$. The rate of input change $\boldsymbol{\Delta U}[k]$ is weighted with the diagonal matrix $\boldsymbol{R}_{\ell_1}$. The weighting matrices $\boldsymbol{Q}_{\ell_1}$ and $\boldsymbol{R}_{\ell_1}$ are defined by (5.3). It is emphasized that these weighting matrices can remain the same as in the non-robust near minimum-time PRG formulation—no re-tuning is necessary.

The cost function (6.3) is expressed depending on $\boldsymbol{U}[k]$ as

$$J^{(l)}_{\ell_1,\mathrm{rob}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) \overset{\substack{(2.47)\\(2.19)}}{=} \left\| \underbrace{\begin{bmatrix} \boldsymbol{Q}_{\ell_1} \mathcal{B}^{(l)}_y \\ \boldsymbol{R}_{\ell_1} \mathcal{B}_{\Delta} \end{bmatrix}}_{\boldsymbol{F}^{(l)}_{\ell_1}} \boldsymbol{U}[k] + \underbrace{\begin{bmatrix} \boldsymbol{Q}_{\ell_1} \left(\mathcal{A}^{(l)}_y \boldsymbol{x}[k] - \boldsymbol{Y}^{\mathrm{r}}[k] \right) \\ \boldsymbol{0}_{N \cdot n_u} \end{bmatrix}}_{\boldsymbol{g}^{(l)}_{\ell_1}[k]} \right\|_1 . \tag{6.4}$$

This cost function is again a convex function, as the ℓ_1-norm of an affine function is convex.

Bringing the input and state constraints (2.25) together with the cost function (6.4) and the min-max formulation for the enumerative scheme of Section 2.3, the optimization problem reads as

$$\begin{aligned} \underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \max_{l \in \{1,2,\ldots,n_{\mathrm{seq}}\}} \left\| \boldsymbol{F}^{(l)}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}^{(l)}_{\ell_1}[k] \right\|_1 \\ \text{subject to} \quad & \mathcal{G} \boldsymbol{U}[k] \leq \mathcal{H}[k]. \end{aligned} \tag{6.5}$$

This optimization problem aims at minimizing the worst-case cost function (*max*) with respect to $\boldsymbol{U}[k]$, while respecting input and state constraints.

In order to eliminate the *max* part, the cost function $\left\| \boldsymbol{F}^{(l)}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}^{(l)}_{\ell_1}[k] \right\|_1$ is minimized *for all* realizations $l \in \{1, 2, \ldots, n_{\mathrm{seq}}\}$ (see Section 2.3.2). Furthermore, to transform the ℓ_1-norm based optimization problem (6.5) to a linear program (LP), the transformation of Section 2.1.5.2 is used. The resulting optimization problem is stated as

$$\begin{aligned} \underset{\boldsymbol{\beta},\, \boldsymbol{U}[k]}{\text{minimize}} \quad & \boldsymbol{1}^{\top} \boldsymbol{\beta} \\ \text{subject to} \quad & -\boldsymbol{\beta} \leq \boldsymbol{F}^{(l)}_{\ell_1} \boldsymbol{U}[k] + \boldsymbol{g}^{(l)}_{\ell_1}[k] \leq \boldsymbol{\beta}, \quad \forall l \in \{1, 2, \ldots, n_{\mathrm{seq}}\}, \\ & \mathcal{G} \boldsymbol{U}[k] \leq \mathcal{H}[k], \end{aligned} \tag{6.6}$$

where the vector $\boldsymbol{\beta}$ is now part of the optimization variables, it is introduced by transforming (6.5) to the LP (6.6). The final transformation to an LP in a standard form that is used by many solvers is given in Section 2.1.5.2. The optimization problem (6.6) is the main part

of the general PRG algorithm presented in Chapter 3—solving this problem in a receding horizon based manner leads to a robust near minimum-time PRG. As it is the case for the robust PRG, the nominal plant model is simulated to shift the prediction horizon if feedback is not available.

The necessary solver for the robust near minimum-time PRG is an LP solver, which is in general computationally less complex than an SOCP solver, which is needed for the robust PRG. As for the robust PRG, robust constraint fulfillment is not considered here.

6.3 Example: Position Tracking Control of a PM DC Motor

The two proposed robust PRG approaches are validated for axis 2 of the positioning system that is described in Chapter 4. Furthermore, the PRGs that are based on RMPC are compared with the non-robust approaches. By comparing the respective control errors, it is shown that the robust performance can indeed be improved through the ideas of enumeration based min-max RMPC.

The experimental evaluation is done for two different reference shapes—a step and a ramp. The use of a ramp shaped reference indicates the possibility of dealing with *arbitrary* varying references. The use of the preview feature of the robust and non-robust PRGs makes it possible to track the reference signals if the constraints are not active. If the constraints are active, it is the task of the PRG to keep the error to the unoptimized reference signal as small as possible.

6.3.1 Tuning of the PRG Approaches

The tuning of the robust PRGs is essentially the same as the tuning of the non-robust approaches of Section 5.3.1. Therefore, an existing PRG can be robustified without the need of re-tuning. This is validated by the experimental results that are presented later; they show that the dynamic performance of the nominal case of the non-robust and the robust approaches is essentially the same. Nevertheless, if the performance of the robust approaches is not satisfying the robust approaches of course allow independent tuning.

Table 6.1 shows an overview of the most important parameters for the PRGs of this chapter. The difference compared with the parameters of Chapter 5 is that the maximum voltage $v_{\max}$ (input constraint) is lowered to 25 V to establish a larger CR (difference between the supply voltage $v_{\mathrm{DC}} = 29.4\,\mathrm{V}$ and $v_{\max} = 25\,\mathrm{V}$). This CR is available for the feedback controller to get active in presence of uncertain parameters if the feedforward voltage is

Maximum voltage $v_{\max}$	$25\,\mathrm{V}$
Maximum current $i_{\max}$	$5\,\mathrm{A}$
$\mathrm{online}_{\mathrm{PRG}}$	false
$\mathrm{feedback}_{\mathrm{PRG}}$	false
preview (reference known in advance)	true
Prediction horizon N	20
Receding horizon shift N_{shift}	1
Standard PRG (robust/non-robust)	
Control error weight $\boldsymbol{q}_{\mathrm{q}}$	$\left[1\,\mathrm{rad}^{-2}\right]$
Rate of input change weight $\boldsymbol{r}_{\mathrm{q}}$	$\left[(0.00005)^2\ \mathrm{V}^{-2}\right]$
Near minimum-time PRG (robust/non-robust)	
Control error weight $\boldsymbol{q}_{\ell_1}$	$\left[1\,\mathrm{rad}^{-1}\right]$
Rate of input change weight $\boldsymbol{r}_{\ell_1}$	$\left[0.0002\,\mathrm{V}^{-1}\right]$

Table 6.1: PRG and robust PRG parameters and system constraints of the SISO reference tracking example

already saturated. A further difference is the use of the preview feature that allows to react to changing reference values *before* the respective change (e.g., a reference step).

6.3.2 Experimental Results

Emulation of uncertain parameters

A perturbed parameter of, for example, $+10\,\%$ is emulated in the control system by changing the respective parameter by $-9.09\,\%$ in the model for the feedback controller design. As the PRGs are operated offline without feedback, the model that is used to simulate the plant in order to shift the prediction horizon is also changed in the same way to emulate a perturbed parameter. This approach is necessary, as it was impracticable to physically change system model parameters.

It is emphasized in Section 2.3 that it is essential that the polytopic uncertainty model encloses *all* system realizations of the linear time-varying (LTV) system that arise in practice. As in this work the parameters of the *continuous-time* model are considered to be uncertain, it is not clear if the polytopic uncertainty model is valid after the zero-order hold (ZOH) discretization is applied to the continuous-time model. This means it is not clear if the convex polytope consisting of the *discrete-time* models for a perturbed parameter of $+10\,\%$ and $-10\,\%$ contains *all* system realizations that lie in between this range (e.g., $+5\,\%$). Neverthe-

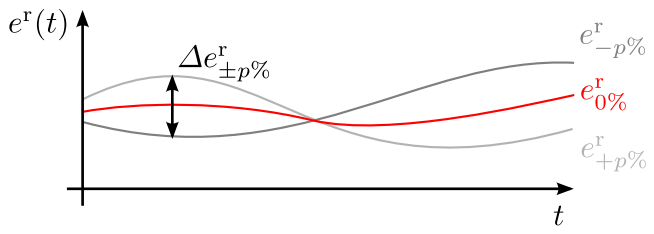


Figure 6.1: Illustration of the control error difference $\Delta e^{\mathrm{r}}_{\pm p\%}$ which is the absolute value of the difference between $e^{\mathrm{r}}_{+p\%}$ (——) and $e^{\mathrm{r}}_{-p\%}$ (——). The nominal control error is denoted by $e^{\mathrm{r}}_{0\%}$ (——).

less, the experimental results show that it is reasonable to neglect the ZOH discretization in terms of the polytopic uncertainty model.

The measurement results that are presented here show an extensive comparison for an uncertain torque constant K_{t} of $\pm 10\,\%$. Summarizing figures show the robust performance for most other system parameters (except J_{m}, J_{l}, and K_{f})[3].

Performance criteria

To evaluate the performance of the proposed approaches, two performance criteria are defined—one to quantify robust performance and another one for dynamic control performance.

The dynamic performance is evaluated with the common integral square error (ISE) criterion

$$C^{\mathrm{ISE}}[k] = T_{\mathrm{s}}^2 \sum_{j=1}^{k} \left(e^{\mathrm{r}}[j]\right)^2, \tag{6.7}$$

where $e^{\mathrm{r}}[k] = \varphi_{\mathrm{l}}^{\mathrm{r}}[k] - \varphi_{\mathrm{l}}[k]$ is the control error.

In order to rate robust performance, the following criterion is introduced

$$C^{\mathrm{ROB}}[k] = T_{\mathrm{s}} \sum_{j=1}^{k} \Delta e^{\mathrm{r}}_{\pm p\%}[j], \tag{6.8}$$

where the control error difference $\Delta e^{\mathrm{r}}_{\pm p\%}[k]$ is defined as

$$\Delta e^{\mathrm{r}}_{\pm p\%}[k] = \left| e^{\mathrm{r}}_{+p\%}[k] - e^{\mathrm{r}}_{-p\%}[k] \right|. \tag{6.9}$$

Figure 6.1 illustrates this robustness criterion. The difference of the control error of two extreme system realizations $\Delta e^{\mathrm{r}}_{\pm p\%}[k]$ ($\pm 10\,\%$ in this work) is considered to represent robust performance, i.e., a small value would mean that the control error is only marginally affected by an uncertain parameter. A prerequisite for the validity of the robust performance

[3] Measurements concerning these parameters do not provide further insights and are therefore omitted.

index $C^{\mathrm{ROB}}[k]$ is that the nominal control error e^{r} is surrounded by the control errors $e^{\mathrm{r}}_{+p\%}$ and $e^{\mathrm{r}}_{-p\%}$, as this is a sign that the polytopic uncertainty model is valid.

Furthermore, the control error $e^{\mathrm{r}}[k]$ and the optimized control error $e^{\mathrm{r,opt}}[k]$ are evaluated (already defined in Section 5.3.4).

Description of exemplary experimental results figures

Figure 6.2a on page 108 serves as an example for a figure that shows the time evolution of important control system signals:

- First plot (from top): Reference load position $\varphi^{\mathrm{r}}_{\mathrm{l}}[k]$ (——), load position $\varphi_{\mathrm{l}}[k]$ (——). The motor position $\varphi_{\mathrm{m}}[k]$ is not shown here.
- Second plot: Control error $e^{\mathrm{r}}[k]$ (——)
- Third plot: Optimized control error $e^{\mathrm{r,opt}}[k]$ (——)
- Fourth plot: Load speed $\omega_{\mathrm{l}}[k]$ (——). The motor speed $\omega_{\mathrm{m}}[k]$ is not shown here.
- Fifth plot: Motor current $i[k]$ (——) and current constraints $\pm i_{\max}$ (- - - -)
- Bottom plot: Motor voltage $v[k]$ (——) and voltage constraints $\pm v_{\max}$ (- - - -). The complete range of the ordinate represents the hard input voltage constraints $\pm v_{\mathrm{DC}}$.

For the nominal cases, the already introduced color scheme is used: Red for load-side values ($\varphi_{\mathrm{l}}[k]$ and $\omega_{\mathrm{l}}[k]$) and values that depend on the load position ($e^{\mathrm{r}}[k]$, $e^{\mathrm{r,opt}}[k]$, etc.); blue is used for the motor-side values ($v[k]$ and $i[k]$). The respective signals for a perturbed parameter are for $+10\,\%$ indicated by (——) and for $-10\,\%$ indicated by (——) (see Figure 6.1). Constraints ($\pm v_{\max}$ and $\pm i_{\max}$) are indicated by black dashed lines (- - - -).

Figure 6.4a on page 110 is an exemplary figure that evaluates the introduced dynamic performance and robust performance criteria:

- Top plot: Control error difference $\Delta e^{\mathrm{r}}_{\pm p\%}[k]$ (——) and its integrated value, the robust performance criterion $C^{\mathrm{ROB}}[k]$ (——)
- Bottom plot: ISE criterion $C^{\mathrm{ISE}}[k]$ for the nominal case (——), for $+10\,\%$ (——), and for $-10\,\%$ (——)

Figure 6.5a on page 110 serves as an example for a figure of the robust performance criterion $C^{\mathrm{ROB}}[k]$ for multiple system parameters and allows evaluating the influence of the different parameters on the robust performance.

Discussion of the experimental results for a reference in the shape of a ramp

Experimental results comparing the non-robust and robust approaches for a reference in the shape of a ramp (slope $30\,\mathrm{rad\,s^{-1}}$) are presented here. The preview feature of the PRGs is exploited to track the reference when the system constraints are not active. An obvious effect of the preview feature is that the optimized reference $\varphi_1^{\mathrm{r,opt}}[k]$ is starting to change *before* the ramp starts.

Figure 6.2 on page 108 shows measurement results for a perturbed torque constant K_t. Comparing the standard PRG (Figure 6.2a) with the robust PRG (Figure 6.2b) shows that the robust performance is improved through the robust PRG formulation: The span of the control error signals $e^\mathrm{r}[k]$ (nominal and $\pm 10\,\%$) of the non-robust approach is much higher than those of the robust approach. The optimized control error $e^{\mathrm{r,opt}}[k]$ is almost zero for the nominal case, which validates the nominal system model, the handling of constraints, and the quality of the applied two-degrees-of-freedom (2-DoF) controller design. It is important to note that the feedback controller is the same for the standard PRG and the robust PRG. The hard input voltage constraints $\pm v_\mathrm{DC} = 29.4\,\mathrm{V}$ are not violated because the voltage constraints $\pm v_\mathrm{max} = 25\,\mathrm{V}$ that are considered in the PRGs introduce a CR for the feedback controller. The current limits (state constraints) are slightly touched here; it can, however, not be guaranteed that these limits are respected. The slight oscillations visible in the control error $e^\mathrm{r}[k]$ before and after the position turn are again caused by the non-uniform weighting of the control error when using quadratic form based PRGs. The preview feature of the PRGs leads to evolutions of the control error $e^\mathrm{r}[k]$ for the nominal case that are nearly symmetric to the time instant where the slope of the reference changes ($t = 0.40\,\mathrm{ms}$ and $t = 0.56\,\mathrm{ms}$).

The experimental results for the near minimum-time PRG and for the robust near minimum-time PRG are presented in Figure 6.3 on page 109. A first observation is that the oscillations in the control error are reduced compared with the quadratic form based PRGs. Nonetheless, the *maximum* control error is increased using the ℓ_1-norm based PRGs, but the control error shows hardly any over- and undershoots. Again, the preview feature of both near-minimum time PRGs leads for the nominal case to control errors $e^\mathrm{r}[k]$ that are symmetric. One of these symmetric responses is visible in the range from $t = 0.34\,\mathrm{ms}$ to $t = 0.45\,\mathrm{ms}$. This time span represents the acceleration phase from zero speed to the desired speed of $30\,\mathrm{rad\,s^{-1}}$. The full voltage range is exploited in the first part of this acceleration phase, where the current is increased. The second part of the acceleration phase, where the current is decreased, does *not* exploit the full voltage range. The reason for this conservative use of the available voltage is the preview feature. Using preview and a long prediction horizon, all PRGs (not only the robust variants) tend to symmetric control

error responses. After the fast current increase at the beginning, a symmetric control error response is achieved through a current trajectory that takes the same time to decrease to zero as it takes the current to increase. However, the voltage that is necessary to decrease the current does not exploit the full voltage range. Hence, it is reasonable to assume that the constant speed that the shape of the reference position commands could be reached faster through a response that is not symmetric. This example illustrates why the ℓ_1-norm based approaches are called *near* minimum-time PRGs. The robust near minimum-time PRG (Figure 6.3b) again shows an improved robust performance compared with the non-robust near minimum-time PRG (Figure 6.3a); the span of the control error signals $e^{\mathrm{r}}[k]$ (nominal and $\pm 10\,\%$) is efficiently reduced for a perturbed torque constant K_{t}. The robust near minimum-time PRG and the non-robust near minimum-time PRG lead to nearly the same nominal responses—this shows that the idea of RMPC based on enumerative schemes is advantageous for the design of robust PRGs.

The criteria for robust performance and for dynamic performance are shown in Figure 6.4 on page 110 for the perturbed torque constant K_{t}. Looking at the end values of the robust performance index $C^{\mathrm{ROB}}[k]$, the robust PRG improves (decreases) this value by over $50\,\%$ compared with the non-robust PRG. The improved robustness is also visible in the C^{ISE} criterion, where the curves for the individual system representations come closer together for the robust PRG. The end-values of $C^{\mathrm{ISE}}[k]$ for the nominal cases of the standard PRG and for the robust PRG are comparable. Hence, the robust PRG shows nearly the same dynamic performance in the nominal case *and* increases the robust performance.

Figure 6.6 on page 111 shows the dynamic performance criterion C^{ISE} and the robust performance criterion $C^{\mathrm{ROB}}[k]$ for the robust and non-robust near minimum-time PRGs. The robust performance for a perturbed torque constant K_{t} is again improved by more than $50\,\%$ through the robust near minimum-time formulation and the curves of $C^{\mathrm{ISE}}[k]$ come closer together. Furthermore, the nominal dynamic performance is not deteriorated by the robust near minimum-time formulation. Comparing the end values of $C^{\mathrm{ISE}}[k]$ of the two ℓ_1-norm based approaches to the two quadratic form based approaches (see Figure 6.4 on page 110), the ℓ_1-norm based PRGs show a worse dynamic behavior according to the ISE criterion. This is not surprising, as the ISE criterion relies—just like the quadratic form based PRGs—on the squared control error.

The robust performance criterion C^{ROB} strongly depends on the uncertain system parameter that is under investigation. Figure 6.5 on page 110 compares the robust performance index $C^{\mathrm{ROB}}[k]$ of the standard PRG with the robust PRG for various system parameters (R, L, K_{t}, c and d). Quite large improvements compared with the standard PRG can be achieved for parameters K_{t} and c, moderate improvements are visible for R and L. For

the damping ratio d the robust performance is slightly decreased when the robust PRG is used. The damping ratio d is a parameter of low importance, as a comparably low value of $C^{\mathrm{ROB}}[k]$ in Figure 6.5 illustrates. The decreased robust performance might be the result of a slight mismatch of the nominal value of d. The shape of the reference also influences robust performance—measurements for a step reference that are shown later confirm this.

Figure 6.7 on page 111 shows the comparison of multiple system parameters for the ℓ_1-norm based near minimum-time and robust near minimum-time approaches. This figure shows quite large robustness improvements for the parameters K_t and R, a moderate improvement is visible for L. The robust performance for c is nearly equal for both approaches. For the damping ratio d, the robust performance is again slightly decreased using the robust near minimum-time PRG.

Discussion of the experimental results for a reference in the shape of a step

Figures 6.8–6.13 on pages 112–115 show the respective evaluations for a reference signal in the shape of a step with a height of 1 mrad. As the basic behavior of the four PRG approaches for a reference in the shape of a ramp has already been evaluated, the discussion here focuses on new insights.

Figures 6.8 and 6.9 on pages 112/113 present the evolution of important control system signals. It is shown that both robust approaches are able to increase the robust performance for an uncertain torque constant K_t. The near minimum-time PRGs show only a small over- and undershoot. The load position $\varphi_\mathrm{l}[k]$ reaches the reference value fast and shows a response that is symmetric to the time instant where the step is commanded ($t = 0.41\,\mathrm{ms}$) because of the preview feature.

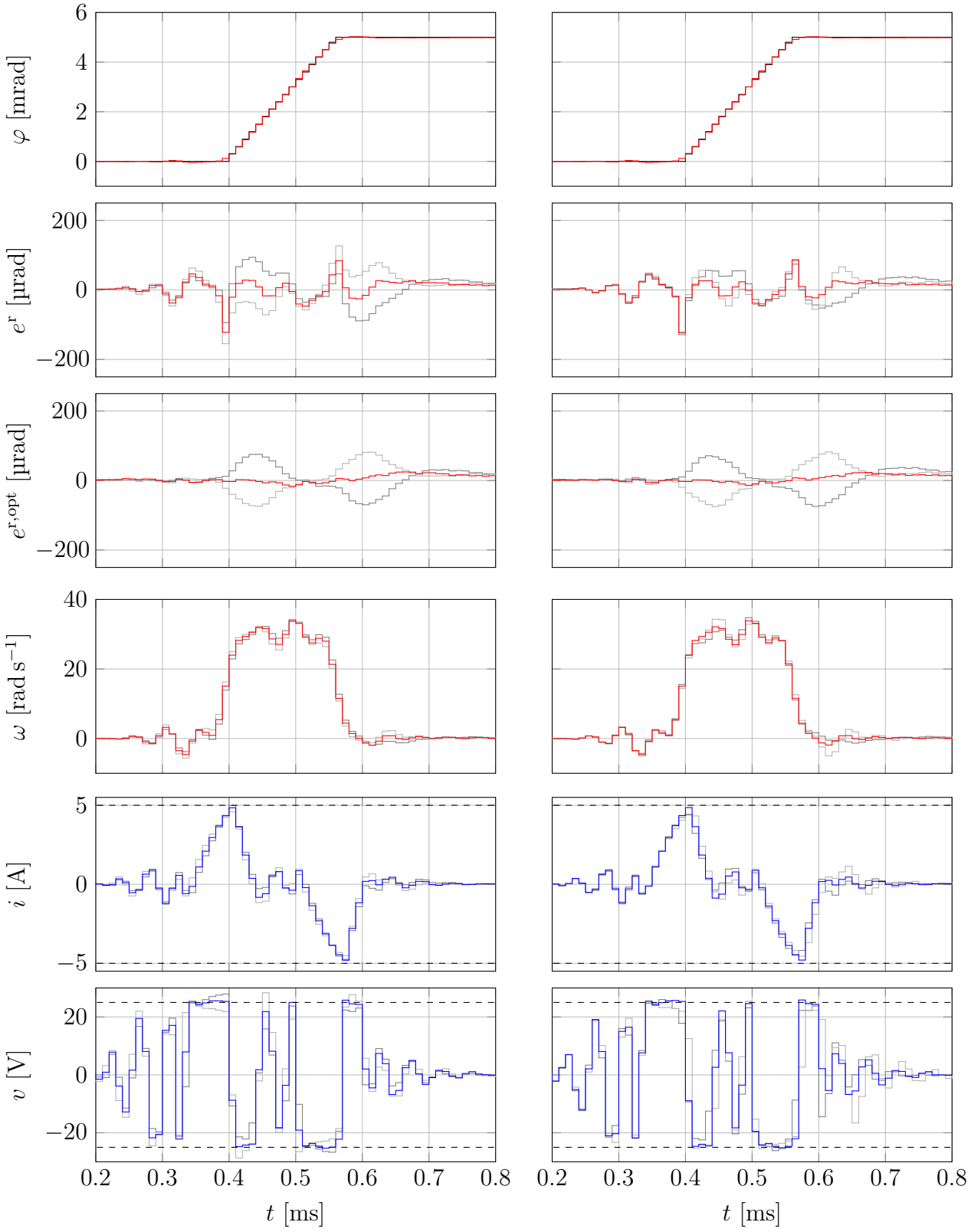
The robust performance evaluation for multiple system parameters for the standard PRG and the robust PRG is shown in Figure 6.11 on page 114. Compared to the reference in the shape of a ramp, the order of the parameters is different. Now, the spring constant c is the most important parameter of the standard PRG approach concerning robust performance. The explanation for this is twofold: First, a reference in the shape of a step has a considerable high-frequency content compared to a ramp reference. Second, the oscillations that are present in the quadratic form based PRGs can easily excite the resonance frequency of the two-mass system (ca. 19.7 kHz) in case of an uncertain spring constant c. Nevertheless, the robust PRG is able to increase the robust performance of most parameters (again except for the damping constant d).

The comparison of multiple parameters for the two near minimum-time approaches is presented in Figure 6.13 on page 115. The two-mass parameter c is not that critical here, as the tendency to show oscillations in the optimized reference is *not* present in the ℓ_1-norm based

PRGs. The robust near minimum-time formulation is again able to achieve more robust results compared with the non-robust formulation (except d).

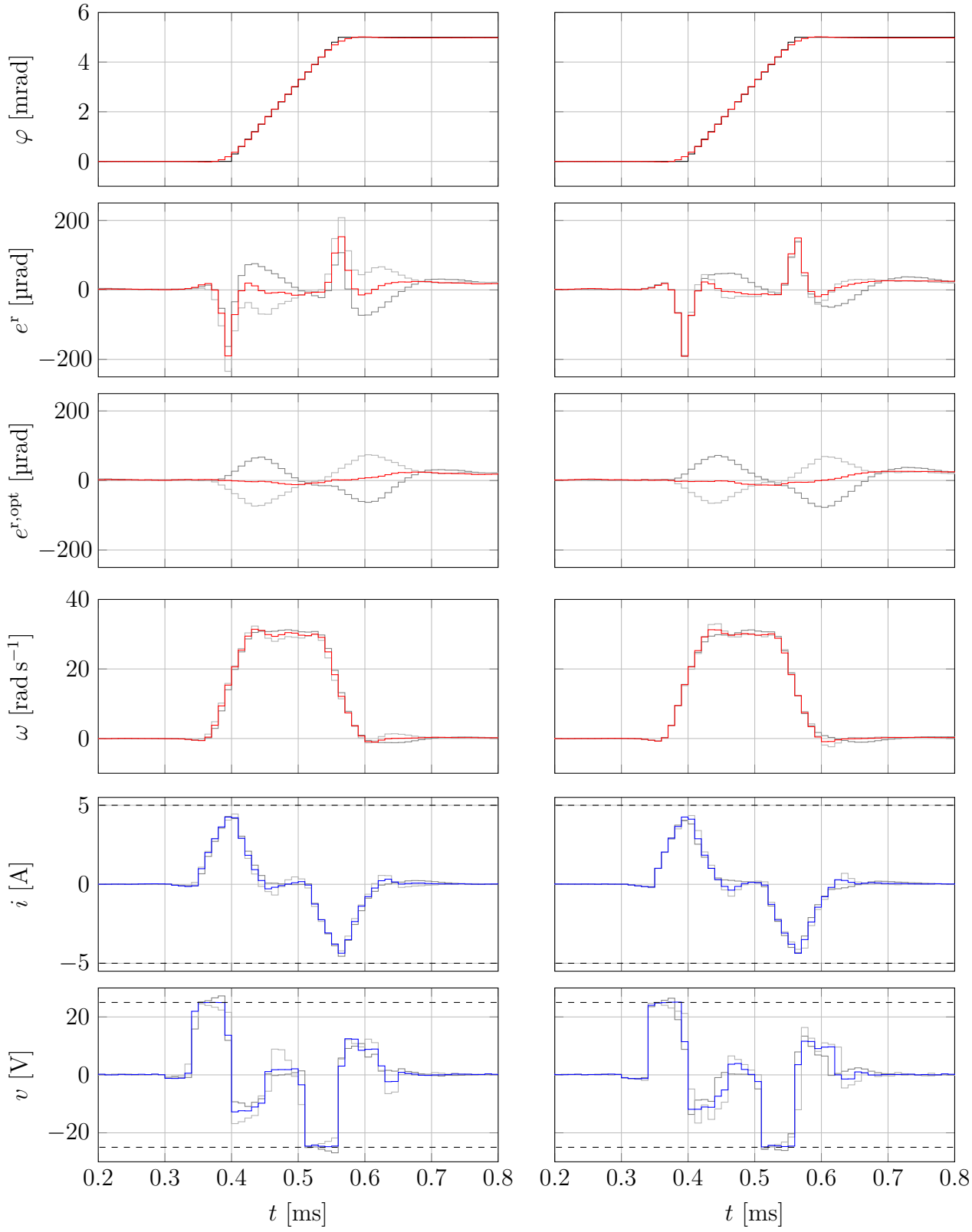
Summary

The experimental results of this section show—at least for the system considered here—that the ideas of RMPC based on enumerative schemes can be transferred to robust PRGs, even if a robust PRG is operated without feedback. The robust PRG and the robust near minimum-time PRG improve the robust performance compared with their respective non-robust counterparts. This increased robust performance is achieved for nearly all system parameters. The ℓ_1-norm based robust near minimum-time PRG is validated to exploit the constraints well and to show less oscillations as well as smaller over- and undershoots compared with the quadratic form based robust PRG. The underlying optimization problem of a robust near minimum-time PRG is an LP, which requires a computationally less complex solver than the robust PRG formulation, which leads to an SOCP. The two reference signals illustrate that the relative importance of the different parameters is dependent on the shape of the reference signal—for example, two-mass related parameters are more important for a step reference, as a step excites the resonance stronger than a ramp. Results for a reference in the shape of a ramp represent a first indication that all presented PRGs can deal with *arbitrary* varying reference signals—more reference shapes are shown in Chapter 7. All results of this section are based on PRGs that makes use of the preview feature, where the reference is known in advance. It is also emphasized that the robust formulations do *not* need re-tuning, as these approaches essentially deliver the same nominal results as their non-robust counterparts. Hence, an existing PRG can be adapted to improve the robust performance—accepting a higher computational effort—without changing the weighting factors. A robust PRG is advantageous if a system parameter is only known up to a certain accuracy or if a system parameter varies over time. Examples for parameters that vary over time are the resistance that varies with temperature or the viscous friction coefficient that is influenced by the aging of the bearings.



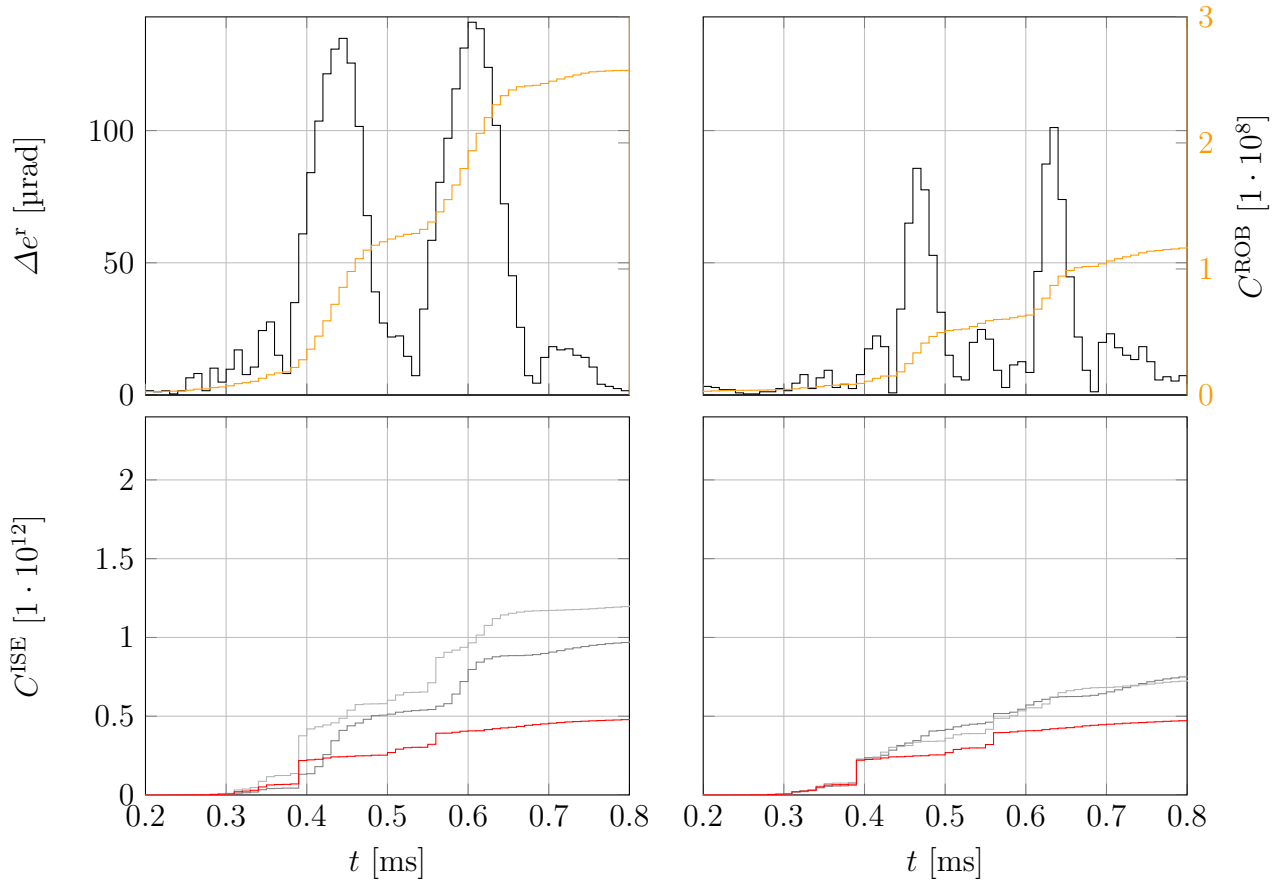
(a) Standard PRG (b) Robust PRG

Figure 6.2: Comparison concerning robust performance for an uncertain torque constant K_{t} ($\pm 10\,\%$) for a reference in the shape of a ramp: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



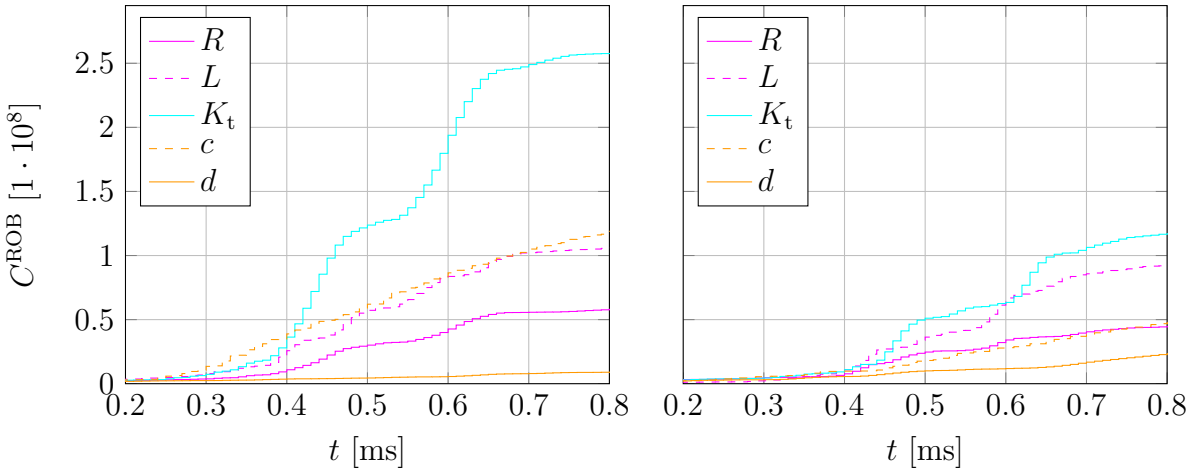
(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.3: Comparison concerning robust performance for an uncertain torque constant K_{t} ($\pm 10\,\%$) for a reference in the shape of a ramp: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)



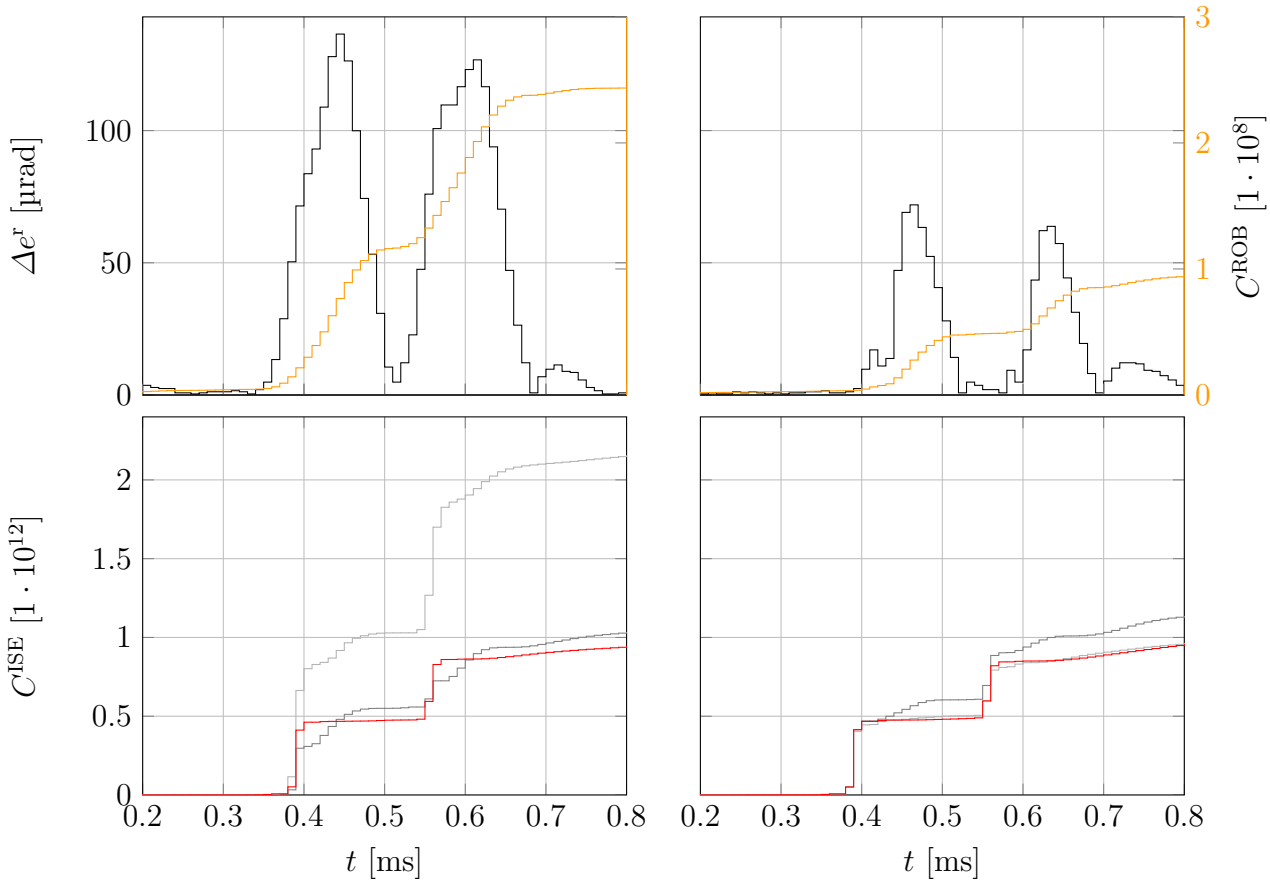
(a) Standard PRG (b) Robust PRG

Figure 6.4: Comparison of robust and dynamic performance criteria for an uncertain torque constant K_{t} ($\pm 10\,\%$) for a reference in the shape of a ramp: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



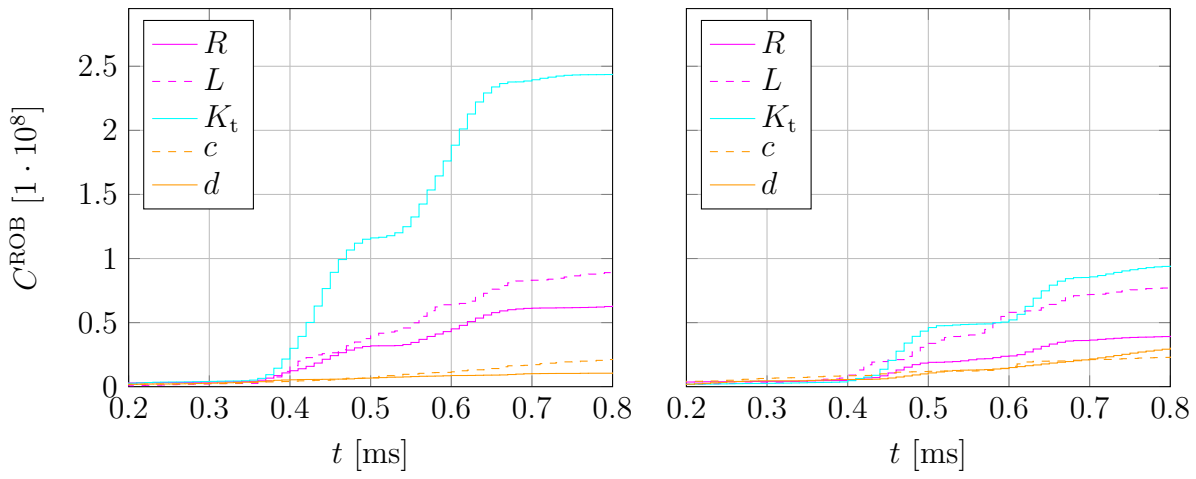
(a) Standard PRG (b) Robust PRG

Figure 6.5: Comparison of robust performance criteria for multiple parameters ($\pm 10\,\%$) for a reference in the shape of a ramp: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



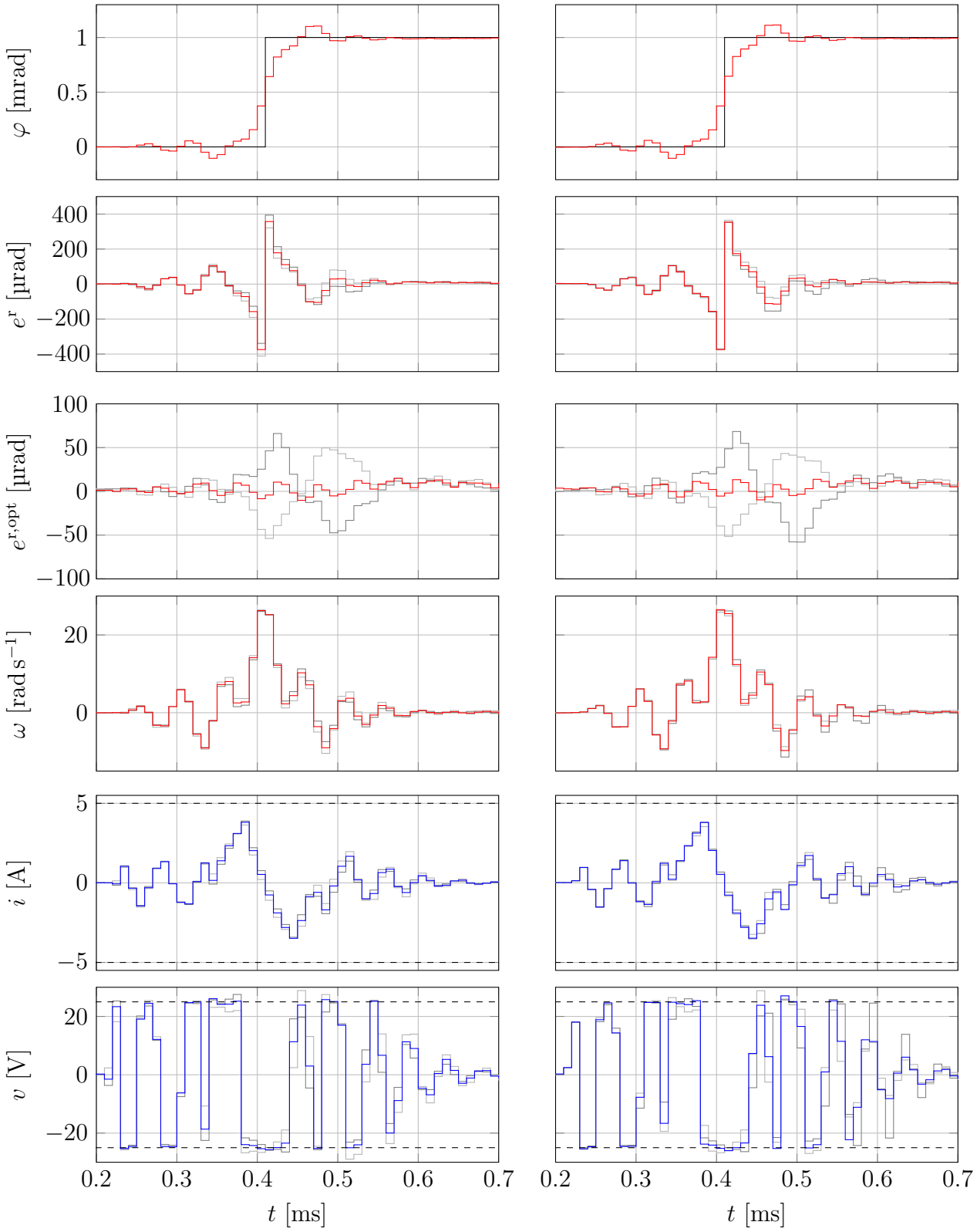
(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.6: Comparison of robust and dynamic performance criteria for an uncertain torque constant K_{t} (±10 %) for a reference in the shape of a ramp: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)



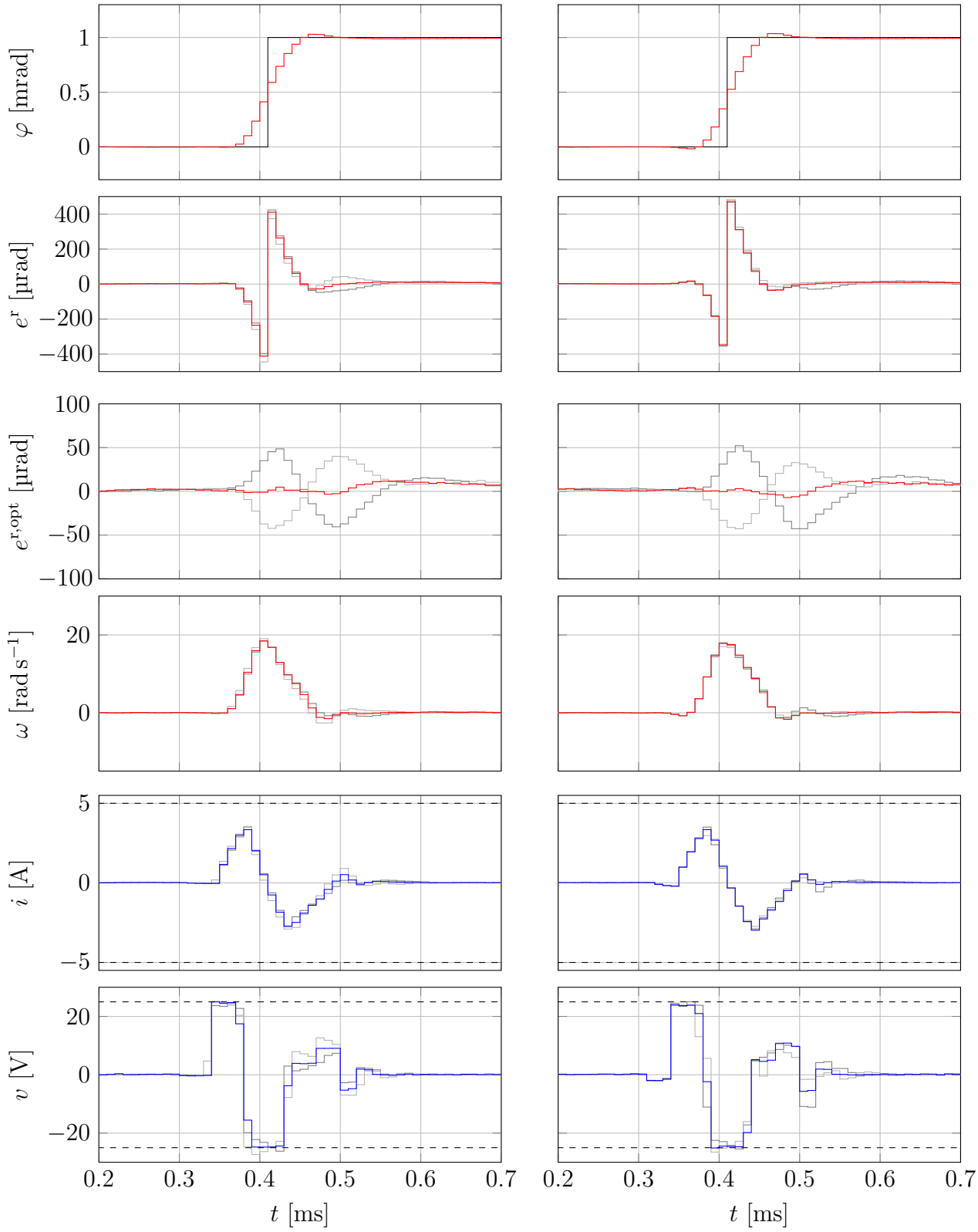
(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.7: Comparison of robust performance criteria for multiple parameters (±10 %) for a reference in the shape of a ramp: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)



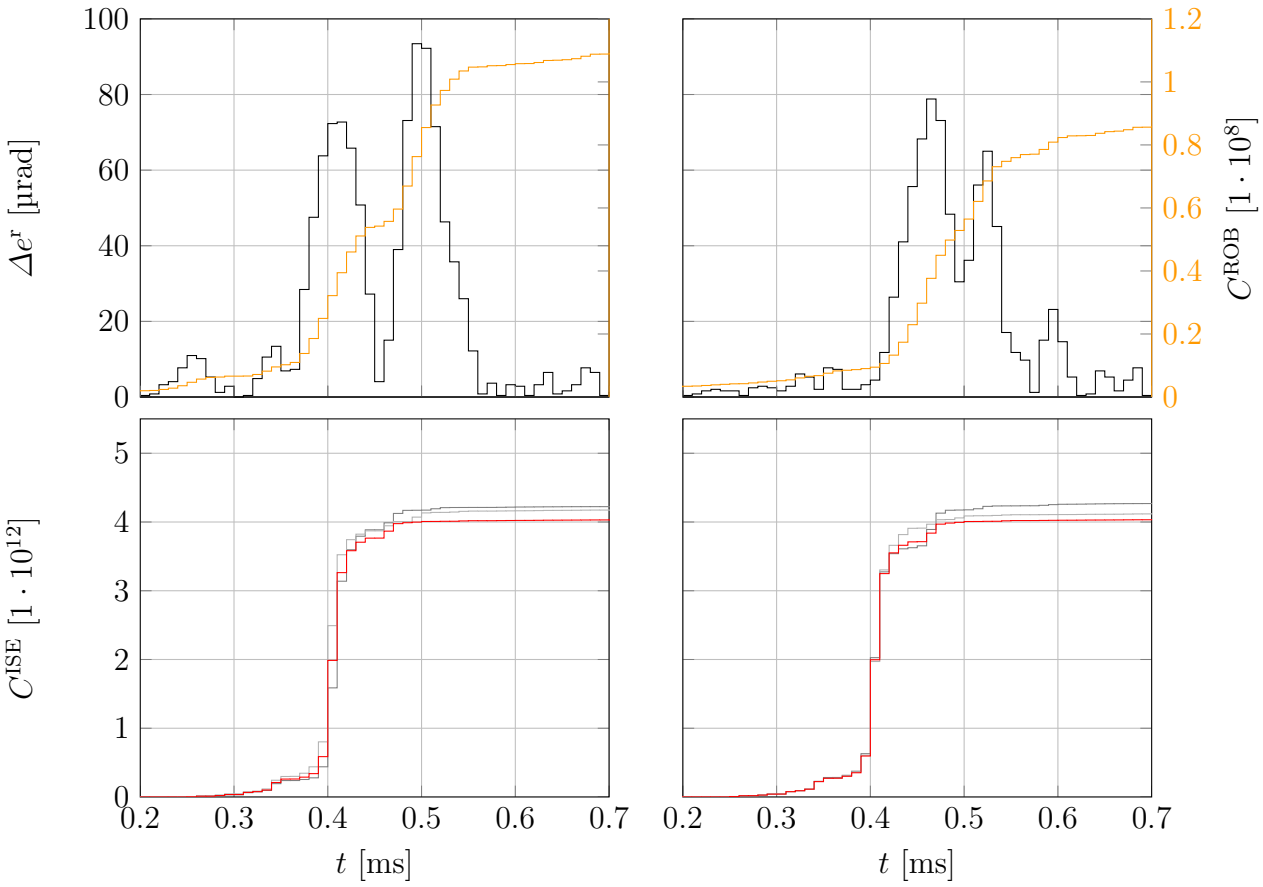
(a) Standard PRG (b) Robust PRG

Figure 6.8: Comparison concerning robust performance for an uncertain torque constant K_t ($\pm 10\,\%$) for a reference in the shape of a step: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



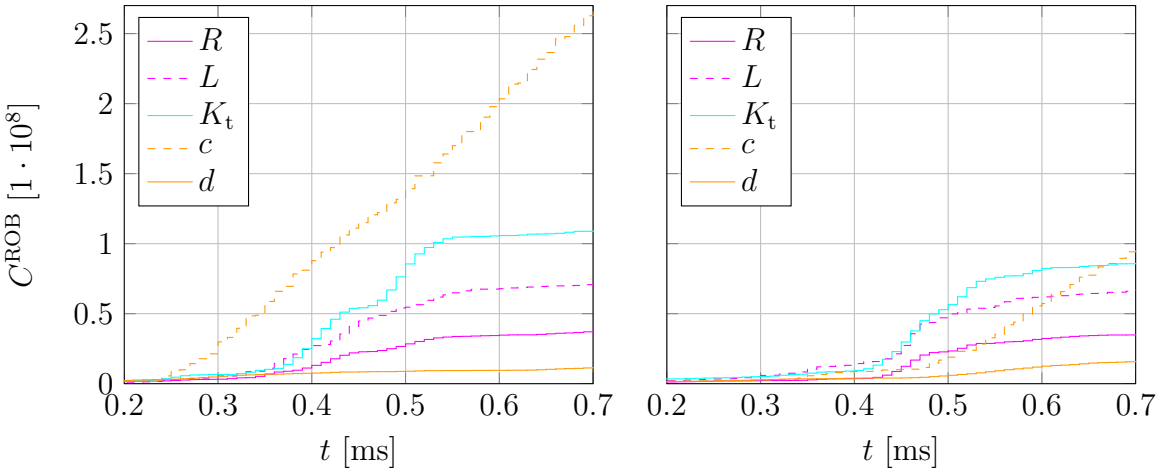
(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.9: Comparison concerning robust performance for an uncertain torque constant K_t ($\pm 10\,\%$) for a reference in the shape of a step: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)



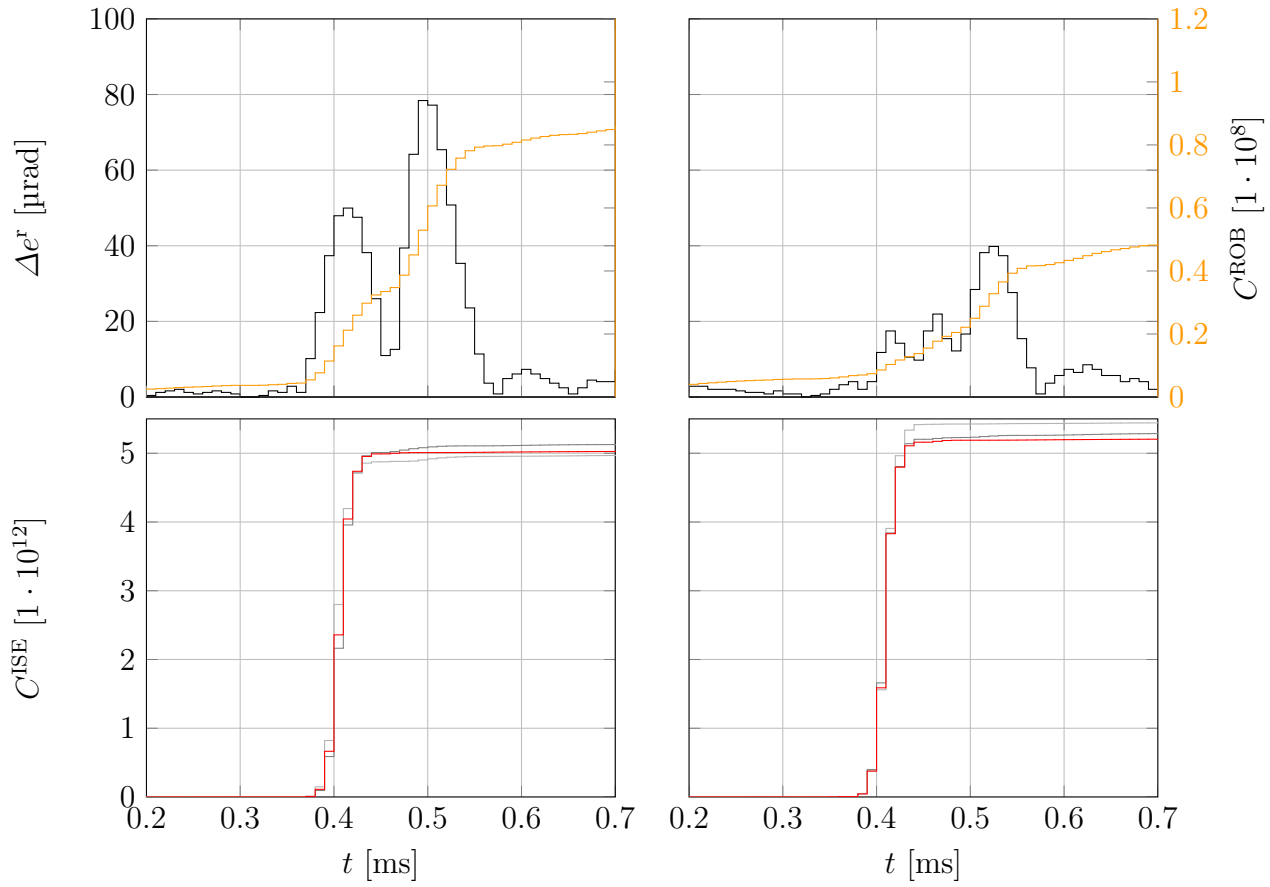
(a) Standard PRG (b) Robust PRG

Figure 6.10: Comparison of robust and dynamic performance criteria for an uncertain torque constant K_{t} ($\pm 10\,\%$) for a reference in the shape of a step: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



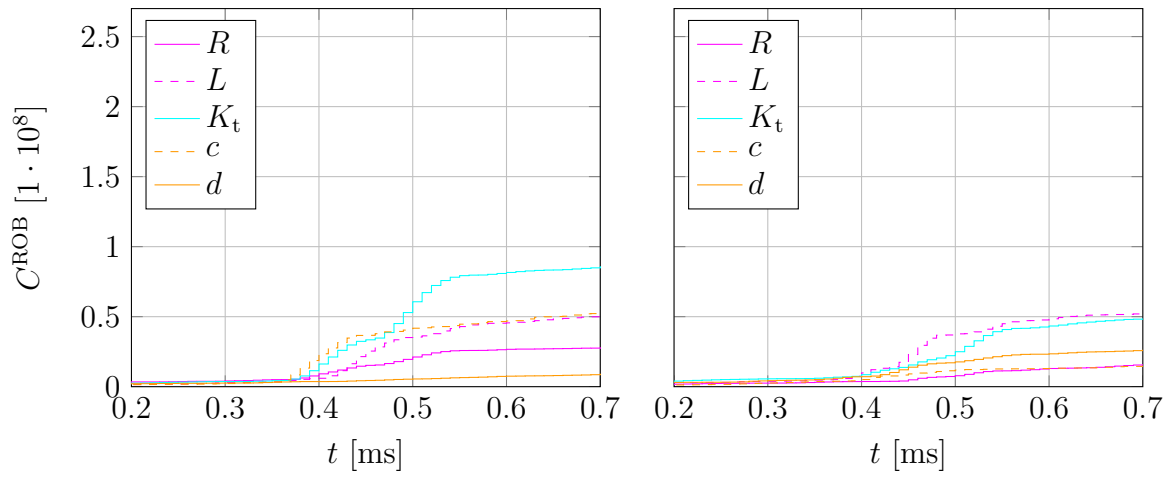
(a) Standard PRG (b) Robust PRG

Figure 6.11: Comparison of robust performance criteria for multiple parameters ($\pm 10\,\%$) for a reference in the shape of a step: Standard PRG vs. robust PRG (see page 102 for the description of the signals)



(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.12: Comparison of robust and dynamic performance criteria for an uncertain torque constant K_{t} ($\pm 10\,\%$) for a reference in the shape of a step: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)



(a) Near minimum-time PRG (b) Robust near minimum-time PRG

Figure 6.13: Comparison of robust performance criteria for multiple parameters ($\pm 10\,\%$) for a reference in the shape of a step: Near minimum-time PRG vs. robust near minimum-time PRG (see page 102 for the description of the signals)

7 Near Minimum-Time PRGs for Constrained MIMO LTI Systems

This chapter emphasizes the ability of predictive reference governors (PRGs) to inherently handle multiple-input, multiple-output (MIMO) systems. A biaxial contouring application consisting of two electric motors (see Chapter 4) is considered. Biaxial contouring aims at following a two-dimensional contour, which is also called path, as close as possible. Furthermore, a predefined path speed should be maintained whenever this is possible.

First, the key cost functions for the standard PRG and the near minimum-time PRG for MIMO systems are given here. After a short introduction to biaxial contouring, two PRGs that minimize the contour error are introduced—one is based on a quadratic form and the other one relies on an ℓ_1-norm based cost function. The contour error is the main performance measure in biaxial contouring. A cost function based on the contour error introduces a virtual coupling between the two motors in order to reduce the contour error. The transformations to optimization problems in standard form are omitted in this chapter, as this topic is extensively treated in previous chapters.

The approaches that are presented here are validated by experimental results for the positioning system that is described in Chapter 4. Two different shapes of a biaxial path are evaluated: One path includes sharp corners and the other path is preprocessed to avoid these sharp corners.

7.1 Standard PRG Formulation

The standard PRG for MIMO systems is based on the general[1] model predictive control (MPC) formulation of Section 2.2.3. The difference to the SISO PRG formulation of Sec-

[1] *General* means that the optimization problems in this work are formulated for MIMO systems. The respective single-input, single-output (SISO) applications lead to a different number of in- and outputs. The basic formulation remains untouched, as SISO systems are a special case of MIMO systems.

tion 5.1 is just the dimension of the in- and outputs. The cost function (5.1) is stated here again for reasons of completeness

$$J_{\mathrm{q}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} (\boldsymbol{Q}_{\mathrm{q}})^{\frac{1}{2}}\, \boldsymbol{E}^{\mathrm{r}}[k] \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}}\, \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2 . \tag{7.1}$$

The principles for a transformation of (7.1), together with the constraints (2.25), to a quadratic program (QP) in standard form can be found in Section 2.2.3. The use of the cost function (7.1) in an MPC scheme for biaxial contouring is called model predictive tracking control (MPTC) (Lam et al., 2013).

7.2 Near Minimum-Time PRG Formulation

The cost function (5.2) of a near minimum-time PRG is repeated here

$$J_{\ell_1}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} \boldsymbol{Q}_{\ell_1} \boldsymbol{E}^{\mathrm{r}}[k] \\ \boldsymbol{R}_{\ell_1} \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_1 . \tag{7.2}$$

Again only the input and output dimensions are changed compared with the PRG formulation of Section 5.2, which is stated generally but is applied to a SISO system. The transformation of (7.2), together with the constraints (2.25), to a linear program (LP) in standard form is stated in Section 2.1.5.2.

7.3 Example: Biaxial Contouring Using Two PM DC Motors

This section introduces the basics of biaxial contouring, including a possibility to approximate the contour error in order to make it usable for the design of PRGs. The PRGs that minimize the contour error (ℓ_1-norm based and quadratic form based) emphasize the flexibility of the PRG cost function design and the applicability to MIMO systems. The four PRG approaches of this chapter are validated by experimental results and compared with each other and with an approach that is considered to be an advanced industry standard.

7.3.1 Introduction to Biaxial Contouring

Biaxial contouring problems often arise in computer numerical control (CNC) machining (Renton and Elbestawi, 2000; Erkorkmaz and Altintas, 2001). The positioning applica-

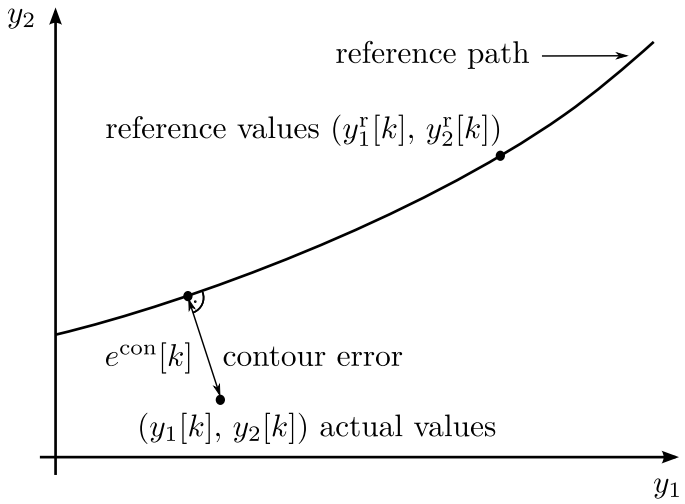


Figure 7.1: Illustration of the contour error $e^{\mathrm{con}}[k]$ for a biaxial system with the axes y_1 and y_2

tion of this work (see Chapter 4) is another example of biaxial contouring. The intention of biaxial contouring is to follow a predefined path (contour) in two dimensions. The main performance measure in these applications is generally the contour error (see Figure 7.1), which is defined as the shortest distance between the actual position and the reference path (Ramesh et al., 2005). The two-dimensional path does *not* contain any timing information. The timing information is introduced by the path speed (also called feedrate), which determines how fast a certain contour is followed. The contour error strongly depends on the path speed—the slower the path is followed, the smaller is the contour error. The reasons for this dependence are constraints of the two motors, which, for example, do not allow changing the acceleration instantly.

Conventional contouring control systems consist of separate controllers for each axis, which reduce the control error of the respective axis. A widespread example of control error minimization is the use of two zero-phase-error tracking control (ZPETC) approaches (Tomizuka, 1987)—the constant phase delay, which is equal for both axes, leads to an improvement of the contouring accuracy. However, separate single axes control approaches do not explicitly consider the contour error. In order to consider the contour error in the control system the cross-coupling controller (CCC), which couples both axes in the control scheme, was introduced by Koren (1980). In the context of contouring control, only a few schemes based on predictive control are known. A generalized predictive control (GPC) scheme for cross-coupling design without constraints is presented in (Zhu and Chen, 2001). Model predictive contouring control (MPCC) including path speed and current constraints is shown in (Lam et al., 2013). However, the non-trivial path parametrization leads to a nonlinear optimization problem with a high computational complexity. An unconstrained MPCC scheme is presented in (Tang and Landers, 2012), which makes it necessary to adapt the path speed

by a supervisory controller in order to maintain constraints and reduce the contour error. Susanu and Dumur (2005) proposed a PRG that minimizes the contour error (based on a quadratic form)—an unconstrained GPC scheme acts as the underlying feedback controller. Another PRG approach that minimizes the contour error was introduced by Chang and Tsao (2014)—this approach relies on a cost function based on a quadratic form and is able to constrain the path speed and the control input. A review of the state of the art in biaxial contouring is presented in (Tang and Landers, 2013).

7.3.1.1 Contour Error Approximation

The contour error, which is defined as the shortest distance between the actual position and the reference path, is traditionally gained through a search process, where shortest distance of the actual position to the reference path is determined. However, as there is no closed-form expression for the contour error, such a search process is unsuitable for the use in control applications. In order to design PRGs that are capable of minimizing the contour error, it is necessary to approximate the contour error through a closed-form expression. This approximation is illustrated in Figure 7.2—the tangent to the reference path at the reference position is used to determine the approximated contour error $\tilde{e}^{\mathrm{con}}[k]$ and the approximated lag distance $\tilde{e}^{\mathrm{lag}}[k]$ (Tang and Landers, 2012; Lam et al., 2013). A system with the output $\boldsymbol{y}[k] = \left[\, y_1[k] \;\; y_2[k] \,\right]^\top$, which contains the two positions of a biaxial positioning system, is considered in the following.

Trigonometric considerations (see Figure 7.2) lead to the approximations

$$\begin{aligned} \tilde{e}^{\mathrm{con}}[k] &= \sin\left(\theta[k]\right)\left(y_1[k] - y_1^{\mathrm{r}}[k]\right) - \cos\left(\theta[k]\right)\left(y_2[k] - y_2^{\mathrm{r}}[k]\right), \\ \tilde{e}^{\mathrm{lag}}[k] &= -\cos\left(\theta[k]\right)\left(y_1[k] - y_1^{\mathrm{r}}[k]\right) - \sin\left(\theta[k]\right)\left(y_2[k] - y_2^{\mathrm{r}}[k]\right), \end{aligned} \tag{7.3}$$

where

$$\theta[k] = \operatorname{atan2}\left(y_2^{\mathrm{r}}[k] - y_2^{\mathrm{r}}[k-1],\, y_1^{\mathrm{r}}[k] - y_1^{\mathrm{r}}[k-1]\right). \tag{7.4}$$

The $\operatorname{atan2}(\cdot,\cdot)$ function returns—in contrast to the standard $\operatorname{atan}(\cdot)$ function—the angle in the correct quadrant by evaluating the signs of both arguments. The subscripts 1 and 2 determine the respective axis of the biaxial system. This approximation makes $\tilde{e}^{\mathrm{con}}[k]$ explicitly dependent on the reference positions $(y_1^{\mathrm{r}}[k],\, y_2^{\mathrm{r}}[k])$ and therefore on time; this is not the case for the exact value $e^{\mathrm{con}}[k]$. This fact makes it possible to include the approximated contour error $\tilde{e}^{\mathrm{con}}[k]$ systematically into a cost function—this would not be possible for the exact value where a search process is necessary. In order to minimize the difference between the exact and the approximated value, the approximated lag distance $\tilde{e}^{\mathrm{lag}}[k]$ has to be minimized

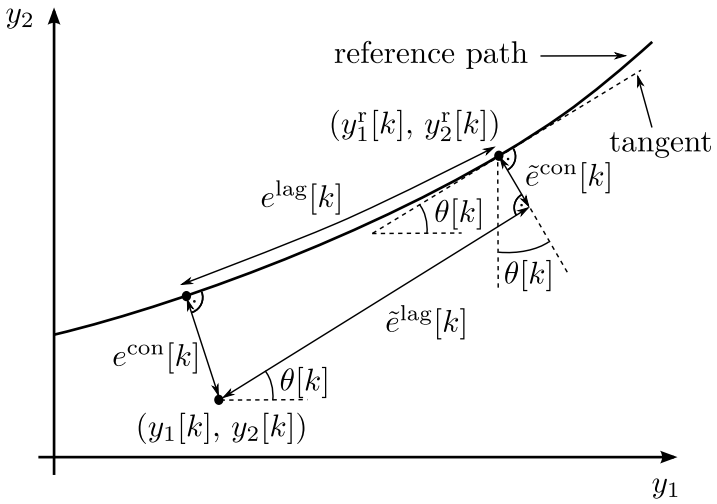


Figure 7.2: Illustration of the contour error $e^{\mathrm{con}}[k]$ and its approximation $\tilde{e}^{\mathrm{con}}[k]$

as well and therefore has to be included in a cost function together with the approximated contour error $\tilde{e}^{\mathrm{con}}[k]$.

The stacked vectors of the approximated contour error $\tilde{\boldsymbol{E}}^{\mathrm{con}}[k]$ and the approximated lag distance $\tilde{\boldsymbol{E}}^{\mathrm{lag}}[k]$ are defined as

$$\begin{aligned}\tilde{\boldsymbol{E}}^{\mathrm{con}}[k] &= \begin{bmatrix}\tilde{e}^{\mathrm{con}}[k+1]^\top & \tilde{e}^{\mathrm{con}}[k+2]^\top & \dots & \tilde{e}^{\mathrm{con}}[k+N]^\top\end{bmatrix}^\top \in \mathbb{R}^N,\\ \tilde{\boldsymbol{E}}^{\mathrm{lag}}[k] &= \begin{bmatrix}\tilde{e}^{\mathrm{lag}}[k+1]^\top & \tilde{e}^{\mathrm{lag}}[k+2]^\top & \dots & \tilde{e}^{\mathrm{lag}}[k+N]^\top\end{bmatrix}^\top \in \mathbb{R}^N.\end{aligned} \tag{7.5}$$

The approximations $\tilde{\boldsymbol{E}}^{\mathrm{con}}[k]$ and $\tilde{\boldsymbol{E}}^{\mathrm{lag}}[k]$ can be calculated as

$$\begin{aligned}\tilde{\boldsymbol{E}}^{\mathrm{con}}[k] &= \boldsymbol{T}^{\mathrm{con}}[k]\,(\boldsymbol{Y}[k] - \boldsymbol{Y}^{\mathrm{r}}[k]) = \boldsymbol{T}^{\mathrm{con}}[k]\boldsymbol{E}^{\mathrm{r}}[k],\\ \tilde{\boldsymbol{E}}^{\mathrm{lag}}[k] &= \boldsymbol{T}^{\mathrm{lag}}[k]\,(\boldsymbol{Y}[k] - \boldsymbol{Y}^{\mathrm{r}}[k]) = \boldsymbol{T}^{\mathrm{lag}}[k]\boldsymbol{E}^{\mathrm{r}}[k],\end{aligned} \tag{7.6}$$

with

$$\begin{aligned}\boldsymbol{T}^{\mathrm{con}}[k] &= \oplus_{i=1}^{N}\boldsymbol{t}^{\mathrm{con}}[k+i] \in \mathbb{R}^{N\times 2\cdot N},\\ \boldsymbol{T}^{\mathrm{lag}}[k] &= \oplus_{i=1}^{N}\boldsymbol{t}^{\mathrm{lag}}[k+i] \in \mathbb{R}^{N\times 2\cdot N},\end{aligned} \tag{7.7}$$

and

$$\begin{aligned}\boldsymbol{t}^{\mathrm{con}}[k] &= \begin{bmatrix}\sin(\theta[k]) & -\cos(\theta[k])\end{bmatrix} \in \mathbb{R}^{1\times 2},\\ \boldsymbol{t}^{\mathrm{lag}}[k] &= \begin{bmatrix}-\cos(\theta[k]) & -\sin(\theta[k])\end{bmatrix} \in \mathbb{R}^{1\times 2}.\end{aligned} \tag{7.8}$$

The matrix vector notation (7.6) of the approximated contour error $\tilde{\boldsymbol{E}}^{\mathrm{con}}[k]$ and the approximated lag distance $\tilde{\boldsymbol{E}}^{\mathrm{lag}}[k]$ is used in the following sections to formulate the PRGs that rely on the contour error.

7.3.2 Contouring PRG Formulation

The contouring PRG presented here includes a cost function based on a quadratic form and is related to the cost function of the MPCC formulation presented in (Tang and Landers, 2012). The cost function for the contouring PRG is defined as

$$J_{\mathrm{q,con}}(\boldsymbol{U}[k], \boldsymbol{x}[k], \boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} \left(\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{con}}\right)^{\frac{1}{2}} \boldsymbol{T}^{\mathrm{con}}[k] \boldsymbol{E}^{\mathrm{r}}[k] \\ \left(\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{lag}}\right)^{\frac{1}{2}} \boldsymbol{T}^{\mathrm{lag}}[k] \boldsymbol{E}^{\mathrm{r}}[k] \\ (\boldsymbol{R}_{\mathrm{q}})^{\frac{1}{2}} \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_2^2 . \tag{7.9}$$

The three main terms of this cost function represent the approximated contour error $\tilde{\boldsymbol{E}}^{\mathrm{con}}[k]$, the approximated lag distance $\tilde{\boldsymbol{E}}^{\mathrm{lag}}[k]$, and the rate of input change $\boldsymbol{\Delta U}[k]$ (from top to bottom), which are multiplied by the respective weighting matrices. The approximated lag distance $\tilde{\boldsymbol{E}}^{\mathrm{lag}}[k]$ also needs to be included in the cost function, as the minimization of this value ensures the validity of the contour error approximation (see Figure 7.2). More specific, a small lag distance leads to a small error between the contour error and its approximation (Tang and Landers, 2012).

The weighting matrices are defined as

$$\begin{aligned} \boldsymbol{Q}_{\mathrm{q}}^{\mathrm{con}} &= \oplus_{i=1}^{N} \mathrm{diag}\{q_{\mathrm{q}}^{\mathrm{con}}\} \in \mathbb{R}^{N \times N}, \\ \boldsymbol{Q}_{\mathrm{q}}^{\mathrm{lag}} &= \oplus_{i=1}^{N} \mathrm{diag}\{q_{\mathrm{q}}^{\mathrm{lag}}\} \in \mathbb{R}^{N \times N}, \\ \boldsymbol{R}_{\mathrm{q}} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\mathrm{q}}\} \in \mathbb{R}^{2 \cdot N \times 2 \cdot N}, \end{aligned} \tag{7.10}$$

where $q_{\mathrm{q}}^{\mathrm{con}} \in \mathbb{R}$, $q_{\mathrm{q}}^{\mathrm{lag}} \in \mathbb{R}$, and $\boldsymbol{r}_{\mathrm{q}} \in \mathbb{R}^2$ are the weights for the contour error, the lag distance, and the rate of input change, respectively. The transformation of (7.9), together with the constraints (2.25), to a QP in standard form is given in Section 2.2.3.

The expressions $\left(\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{con}}\right)^{\frac{1}{2}} \boldsymbol{T}^{\mathrm{con}}[k]$ and $\left(\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{lag}}\right)^{\frac{1}{2}} \boldsymbol{T}^{\mathrm{lag}}[k]$ in (7.9) can be seen as time-varying weighting matrices, in contrast to time independent weighting matrices that are used in the PRGs that do not rely on the contour error. These time-varying weighting matrices introduce a virtual coupling between the two axes in order to minimize the contour error. By setting $q_{\mathrm{q}}^{\mathrm{con}} = q_{\mathrm{q}}^{\mathrm{lag}}$ and using the trigonometric identity $(\sin(\theta[k]))^2 + (\cos(\theta[k]))^2 = 1$, the contouring PRG cost function (7.9) would reduce to the standard PRG cost function (7.1).

7.3.3 Near Minimum-Time Contouring PRG Formulation

As in the previous chapters, an ℓ_1-norm based PRG is introduced here to achieve near minimum-time control results with reduced over- and undershoots. The PRG proposed here is called near minimum-time contouring PRG and is based on the cost function

$$J_{\ell_1,\mathrm{con}}(\boldsymbol{U}[k],\boldsymbol{x}[k],\boldsymbol{Y}^{\mathrm{r}}[k]) = \left\| \begin{bmatrix} \boldsymbol{Q}_{\ell_1}^{\mathrm{con}} \boldsymbol{T}^{\mathrm{con}}[k] \boldsymbol{E}^{\mathrm{r}}[k] \\ \boldsymbol{Q}_{\ell_1}^{\mathrm{lag}} \boldsymbol{T}^{\mathrm{lag}}[k] \boldsymbol{E}^{\mathrm{r}}[k] \\ \boldsymbol{R}_{\ell_1} \boldsymbol{\Delta U}[k] \end{bmatrix} \right\|_1 . \tag{7.11}$$

The ℓ_1-norm based cost function (7.11) consists of the same terms as the quadratic form based contouring PRG cost function (7.9).

The weighting matrices are defined as

$$\begin{aligned} \boldsymbol{Q}_{\ell_1}^{\mathrm{con}} &= \oplus_{i=1}^{N} \mathrm{diag}\{q_{\ell_1}^{\mathrm{con}}\} \in \mathbb{R}^{N\times N}, \\ \boldsymbol{Q}_{\ell_1}^{\mathrm{lag}} &= \oplus_{i=1}^{N} \mathrm{diag}\{q_{\ell_1}^{\mathrm{lag}}\} \in \mathbb{R}^{N\times N}, \\ \boldsymbol{R}_{\ell_1} &= \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\ell_1}\} \in \mathbb{R}^{2\cdot N\times 2\cdot N}, \end{aligned} \tag{7.12}$$

where $q_{\ell_1}^{\mathrm{con}} \in \mathbb{R}$, $q_{\ell_1}^{\mathrm{lag}} \in \mathbb{R}$, and $\boldsymbol{r}_{\ell_1} \in \mathbb{R}^2$ are the weights for the contour error, the lag distance, and the rate of input change, respectively. The transformation of (7.11) together with the constraints (2.25) to an LP in standard form is given in Section 2.1.5.2.

7.3.4 Tuning and System Model of the PRG Approaches

The contouring application treated here consists of two motors which leads to a MIMO system with two inputs $n_u = 2$ and two outputs $n_y = 2$. Considering these dimensions, the weights of the standard PRG ($\boldsymbol{q}_{\mathrm{q}}$ and $\boldsymbol{r}_{\mathrm{q}}$) and of the near minimum-time PRG ($\boldsymbol{q}_{\ell_1}$ and $\boldsymbol{r}_{\ell_1}$) are two-dimensional vectors. The tuning principle remains the same as presented for the SISO case in Section 5.3.1—except that the tuning theoretically needs to be done separately for each axis. Nevertheless, as the system consists of motors that only differ in their two-mass behavior, the weights for both axes are chosen to be equal (see Table 7.1). More specific, the scalar weights of the SISO case remain unchanged and are just extended to two-dimensional vectors.

The weights for the contouring PRG ($q_{\mathrm{q}}^{\mathrm{con}}$, $q_{\mathrm{q}}^{\mathrm{lag}}$, and $\boldsymbol{r}_{\mathrm{q}}$) and the near minimum-time contouring PRG ($q_{\ell_1}^{\mathrm{con}}$, $q_{\ell_1}^{\mathrm{lag}}$, and $\boldsymbol{r}_{\ell_1}$) require some more attention. The weights of the rate of input change $\boldsymbol{r}_{\mathrm{q}}$ and $\boldsymbol{r}_{\ell_1}$ are chosen to be same as for the standard PRG and the near minimum-time PRG. The lag weights are chosen to be $q_{\mathrm{q}}^{\mathrm{lag}} = 1\,\mathrm{rad}^{-2}$ and $q_{\ell_1}^{\mathrm{lag}} = 1\,\mathrm{rad}^{-1}$, as one term of the three cost function terms can be chosen arbitrarily without loss of generality. The remaining contour error weights $q_{\mathrm{q}}^{\mathrm{con}}$ and $q_{\ell_1}^{\mathrm{con}}$ have to be chosen to achieve the desired

contouring accuracy—an increased value usually leads to a decreased contour error. The heuristically determined weights are shown in Table 7.1.

As in the PRG applications of the other chapters, the PRGs presented here are executed offline ($\text{online}_{\text{PRG}} = \text{false}$) and therefore no feedback information can be used ($\text{feedback}_{\text{PRG}} = \text{false}$). Just like for the robust approaches, the preview feature, where the reference is known in advance, is used here. The MIMO PRGs of this chapter use a receding horizon shift of $N_{\text{shift}} = 5$ which reduces the calculation time for the whole reference trajectory by a factor of five compared with $N_{\text{shift}} = 1$ (offline operation of the PRGs).

The PRG approaches of this chapter require a single state-space model that incorporates both axes of the positioning system of Chapter 4. To achieve this, the discrete-time state-space model (4.9) is equipped with the following state-space matrices

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0}_{5\times5} \\ \boldsymbol{0}_{5\times5} & \boldsymbol{A}_2 \end{bmatrix}, \; \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_1 & \boldsymbol{0}_{5\times1} \\ \boldsymbol{0}_{5\times1} & \boldsymbol{B}_2 \end{bmatrix}, \; \boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_1 & \boldsymbol{0}_{1\times5} \\ \boldsymbol{0}_{1\times5} & \boldsymbol{C}_2 \end{bmatrix}, \tag{7.13}$$

where $\boldsymbol{A} \in \mathbb{R}^{10\times10}$ is the state matrix, $\boldsymbol{B} \in \mathbb{R}^{10\times2}$ is the input matrix, and $\boldsymbol{C} \in \mathbb{R}^{2\times10}$ denotes the output matrix. The subscripts 1 and 2 identify the respective axis of the biaxial system. The respective state, input and output vectors are

$$\boldsymbol{x}[k] = \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix}, \; \boldsymbol{u}[k] = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \; \boldsymbol{y}[k] = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}. \tag{7.14}$$

7.3.5 Acceleration- and Jerk-limited Trajectory Planning: An Advanced Industry Standard

The advanced industry standard presented in Section 5.3.2 is also used here for the biaxial contouring example to compare the presented PRGs with an approach that is well known in industry. The acceleration and the jerk are limited separately for both axes of the biaxial positioning system in order to account for the respective voltage and current constraints. The advanced industry standard does not deliver feedforward signals that can be used for dynamic, model-based feedforward control (DynFF). Hence, it is necessary to use ZPETC in the two-degrees-of-freedom (2-DoF) based control structure. The use of ZPETC introduces a constant delay of two sampling instants for the application considered here—this delay is compensated to allow a fair comparison with the other approaches concerning the control error and the optimized control error.

Maximum voltage $v_{\max}$	$28\,\mathrm{V}$
Maximum current $i_{\max}$	$5\,\mathrm{A}$
$\mathrm{online}_{\mathrm{PRG}}$	false
$\mathrm{feedback}_{\mathrm{PRG}}$	false
preview (reference known in advance)	true
Prediction horizon N	20
Receding horizon shift N_{shift}	5
Standard PRG	
Control error weight $\boldsymbol{q}_{\mathrm{q}}$	$\left[1\,\mathrm{rad}^{-2} \quad 1\,\mathrm{rad}^{-2}\right]^{\top}$
Rate of input change weight $\boldsymbol{r}_{\mathrm{q}}$	$\left[(0.00005)^2\,\mathrm{V}^{-2} \quad (0.00005)^2\,\mathrm{V}^{-2}\right]^{\top}$
Near minimum-time PRG	
Control error weight $\boldsymbol{q}_{\ell_1}$	$\left[1\,\mathrm{rad}^{-1} \quad 1\,\mathrm{rad}^{-1}\right]^{\top}$
Rate of input change weight $\boldsymbol{r}_{\ell_1}$	$\left[0.0002\,\mathrm{V}^{-1} \quad 0.0002\,\mathrm{V}^{-1}\right]^{\top}$
Contouring PRG	
Contour error weight $q_{\mathrm{q}}^{\mathrm{con}}$	$(100)^2\,\mathrm{rad}^{-2}$
Lag weight $q_{\mathrm{q}}^{\mathrm{lag}}$	$1\,\mathrm{rad}^{-2}$
Rate of input change weight $\boldsymbol{r}_{\mathrm{q}}$	$\left[(0.00005)^2\,\mathrm{V}^{-2} \quad (0.00005)^2\,\mathrm{V}^{-2}\right]^{\top}$
Near minimum-time contouring PRG	
Contour error weight $q_{\ell_1}^{\mathrm{con}}$	$100\,\mathrm{rad}^{-1}$
Lag weight $q_{\ell_1}^{\mathrm{lag}}$	$1\,\mathrm{rad}^{-1}$
Rate of input change weight $\boldsymbol{r}_{\ell_1}$	$\left[0.0002\,\mathrm{V}^{-1} \quad 0.0002\,\mathrm{V}^{-1}\right]^{\top}$

Table 7.1: PRG parameters and system constraints of the biaxial contouring example

7.3.6 Experimental Results

Measurement results for the MIMO PRG approaches and for the advanced industry standard applied to the positioning system of Chapter 4 are presented in this section. The results are shown for a path that includes sharp corners and a preprocessed path that avoids these sharp corners. *Preprocessed* means in this context that the reference, which is fed to the PRGs, already respects given acceleration and jerk limits.

The parts of the control structures of Figures 3.1 and 4.6 that are working online are implemented separately for each axis. These online parts represent the 2-DoF based controllers. Nevertheless, the PRGs deal with the MIMO system that is described by the matrices (7.13).

Performance criteria

The main performance measure that is evaluated here is the *exact* value of the contour error $e^{\mathrm{con}}[k]$, which is gained through a search process. The approximated contour error, which is used for the formulation of the PRG approaches that minimize the contour error, is not shown here—a validation of this approximation is given in (Dötlinger and Kennel, 2013b).

Other performance measures are the control error $e^{\mathrm{r}}[k]$ and the optimized control error $e^{\mathrm{r,opt}}[k]$, which are defined in Section 5.3.4.

Furthermore, the path speed is evaluated. The path speeds for the motor $\omega_m^{\mathrm{path}}[k]$ and the load $\omega_l^{\mathrm{path}}[k]$ are defined as the Euclidean norms of the respective speeds of the two single axes

$$\begin{aligned} \omega_m^{\mathrm{path}}[k] &= \sqrt{\left(\omega_{\mathrm{m},1}[k]\right)^2 + \left(\omega_{\mathrm{m},2}[k]\right)^2}, \\ \omega_l^{\mathrm{path}}[k] &= \sqrt{\left(\omega_{\mathrm{l},1}[k]\right)^2 + \left(\omega_{\mathrm{l},2}[k]\right)^2}. \end{aligned} \tag{7.15}$$

Description of exemplary experimental results figures

Figures 7.3a and 7.3b on page 131 serve as example figures that show the time evolution of important control system signals for the axes 1 and 2:

- First plot (from top): Reference load position $\varphi_{\mathrm{l}}^{\mathrm{r}}[k]$ (——), load position $\varphi_{\mathrm{l}}[k]$ (——), motor position $\varphi_{\mathrm{m}}[k]$ (——)
- Second plot: Control error $e^{\mathrm{r}}[k]$ (——)
- Third plot: Optimized control error $e^{\mathrm{r,opt}}[k]$ (——)
- Fourth plot: Load speed $\omega_{\mathrm{l}}[k]$ (——), motor speed $\omega_{\mathrm{m}}[k]$ (——)
- Fifth plot: Motor current $i[k]$ (——) and current constraints $\pm i_{\max}$ (- - - -)
- Bottom plot: Motor voltage $v[k]$ (——) and voltage constraints $\pm v_{\max}$ (- - - -). The complete range of the ordinate represents the hard input voltage constraints $\pm v_{\mathrm{DC}}$.

Figure 7.3c on page 131 exemplary shows the two-dimensional (biaxial) reference path (——), the load position path (——), and the motor position path (——).

Figure 7.3d on page 131 evaluates the contour error $e^{\mathrm{con}}[k]$ (——).

Figure 7.3e on page 131 is an exemplary figure that shows the path speed for the load $\omega_l^{\mathrm{path}}[k]$ (——) and the motor $\omega_m^{\mathrm{path}}[k]$ (——).

Discussion of the experimental results for contouring of a path with sharp corners

The results presented here are based on a path that exhibits sharp corners and is therefore quite challenging for biaxial contouring. Furthermore, the path contains three quarters of a circle and a diagonal line segment; the shape is therefore related to the ISO standards for machine tools (ISO 230-4:2005, 2005; ISO 230-6:2002, 2002; Weck and Brecher, 2006). The two reference trajectories for the respective axes/motors are gained through constant path speed interpolation, where the reference trajectories are determined such that a constant path speed is achieved. Hence, sharp corners in the path lead to speed changes in the shape of a step for the respective axis. Due to these step like speed changes as well as the voltage and current constraints of the motors, the reference path cannot be exactly followed by the advanced industry standard, the standard PRG, and the near minimum-time PRG. The PRGs that minimize the contour error, however, make it possible to follow the path almost perfectly through the coupling of both axes, which is introduced by the contour error minimization. The two-dimensional path was transferred to the respective reference trajectories through a constant path speed of $\omega_l^{\mathrm{path}} = 30\,\mathrm{rad\,s^{-1}}$.

Figure 7.3 on page 131 shows experimental results for the advanced industry standard. The voltage constraints are violated for short amounts of time due to the choice of the maximum jerk according to approach (B) of Section 5.3.2, where a trade-off is found between a fast response and the handling of constraints. Additionally, the neglected two-mass behavior and the neglected back electromotive force (EMF) for the choice of the maximum jerk might be reasons for exceeding the voltage constraints. The current constraints are only slightly violated. The limitation of the acceleration and the jerk leads to a relatively high contour error $e^{\mathrm{con}}[k]$ and a decreased path speed in the two sharp corners. The evolution of the optimized control error $e^{\mathrm{r,opt}}[k]$ exhibits values that are a few times higher than the noise level. These comparably high values result from modeling errors and from the use of a ZPETC based 2-DoF approach that exhibits non-unity gain for high frequencies. The PRG approaches presented later make use of DynFF and therefore show smaller values in the evolution of the optimized control error $e^{\mathrm{r,opt}}[k]$. The plot of the contour error shows that there are nearly no over- and undershoots in the corners.

The results for the standard PRG approach that are presented in Figure 7.4 on page 132 show—compared with the advanced industry standard—decreased maximum values of the contour error $e^{\mathrm{con}}[k]$. The voltage and current constraints are respected and fully exploited. However, the evolutions of the contour error and the control errors show the typical over- and undershoots of the quadratic form based cost functions.

Figure 7.5 on page 133 presents the results for the near minimum-time PRG. The peak values of the contour error $e^{\mathrm{con}}[k]$ lie in between the values of the advanced industry stan-

dard and the standard PRG. However, compared with the standard PRG, the over- and undershoots are almost completely removed—a behavior that is also shown by the near minimum-time formulations of the previous chapters. Furthermore, the evolution of the input voltage contains less high-frequency content, which is advantageous as unmodeled higher order dynamics are involved here.

The experimental results for the contouring PRG and the near minimum-time contouring PRG are shown in Figures 7.6 and 7.7 on pages 134/135, respectively. Both approaches can reduce the contour error $e^{\mathrm{con}}[k]$ to a value that is in the range of the optimized control error $e^{\mathrm{r,opt}}[k]$. Hence, it is likely that the contour error evolutions of both PRGs mainly result from modeling errors. Nevertheless, both approaches enable following the path with sharp corners almost perfectly; this is a result of the coupling of the axes, which is introduced by the contour error minimization. In order to follow the path even in the corners a path speed of zero is necessary—this is shown in the plot of the path speed. In order to regain the overall timing with respect to the reference positions, the path speed increases before and after the sharp corners. In the regions of the sharp corners this timing is lost in order to minimize the contour error $e^{\mathrm{con}}[k]$—the high values of the control errors $e^{\mathrm{r}}[k]$ confirm this—nevertheless, the contour error, which is the performance measure here, is efficiently minimized. The biggest difference between the contouring PRG and the near minimum-time contouring PRG is the shape of the input voltage $v[k]$. The considerable high-frequency content of the contouring PRG leads to high-frequency oscillation in the optimized control error because of an unmodeled higher order resonance. The evolution of the input voltage $v[k]$ of the near minimum-time contouring PRG is smoother.

The three quarters of a circle that are also present in the two-dimensional path again emphasize the two-mass behavior. The motor position $\varphi_{\mathrm{m}}[k]$ clearly differs from the load position $\varphi_{\mathrm{l}}[k]$. This can be explained by the gain difference that the respective transfer functions show when leaving the low-frequency range (see Bode plots in Figures 4.8 and 4.10 on pages 66/67). Furthermore, both positions differ especially in the corners.

Discussion of the experimental results for contouring of a preprocessed path

The experimental results for a preprocessed path demonstrate the advantages of the contouring PRGs for paths that do not show sharp corners. The two-dimensional path used here (see, e.g., Figure 7.8 on page 137) was preprocessed using the idea of acceleration and jerk limitation; this approach is considered here being an advanced industry standard. This means that the sharp corners of the path are replaced by curvatures[2]—this should allow the underlying control system to follow the path without violating the constraints. The path

[2]The sharp corner that is visible at the origin of the two-dimensional path is the result of the starting and end point of the path.

speed is set to $\omega_l^{\text{path}} = 60\,\text{rad}\,\text{s}^{-1}$, which is twice the path speed of the contouring results for sharp corners. The respective reference position trajectories are gained through constant path velocity interpolation. As the basic results for all five approaches are discussed in the previous section, this section mainly deals with new insights.

The experimental results were gained for an acceleration that corresponds to a current limit of $\pm 7.5\,\text{A}$. In order to show the contouring capability of the presented contouring PRG and the near minimum-time contouring PRG, their current limit remains unchanged at $\pm 5\,\text{A}$. Hence, the preprocessed path exhibits high curvatures that traditionally cannot be followed if the current limit of $\pm 5\,\text{A}$ needs to be respected. This means that an existing trajectory planning scheme—acceleration- and jerk-limited in this case—can be equipped with limits that lead to high curvatures, which is advantageous for precise positioning. A PRG that minimizes the contour error can then be used to follow the two-dimensional path.

Figure 7.8 on page 137 shows the results for the acceleration- and jerk-limited trajectories with a current limit of $\pm 7.5\,\text{A}$. ZPETC is used to track the references. A logical result is that the current limit that is valid for the PRG approaches ($\pm 5\,\text{A}$) is violated. The voltage constraints are again exceeded for some sampling instants due to the choice of the maximum jerk according to approach (B) (see Section 5.3.2). Nevertheless, the evolution of the contour error $e^{\text{con}}[k]$ shows comparably small values, as the preprocessed path mostly respects the hard voltage constraints.

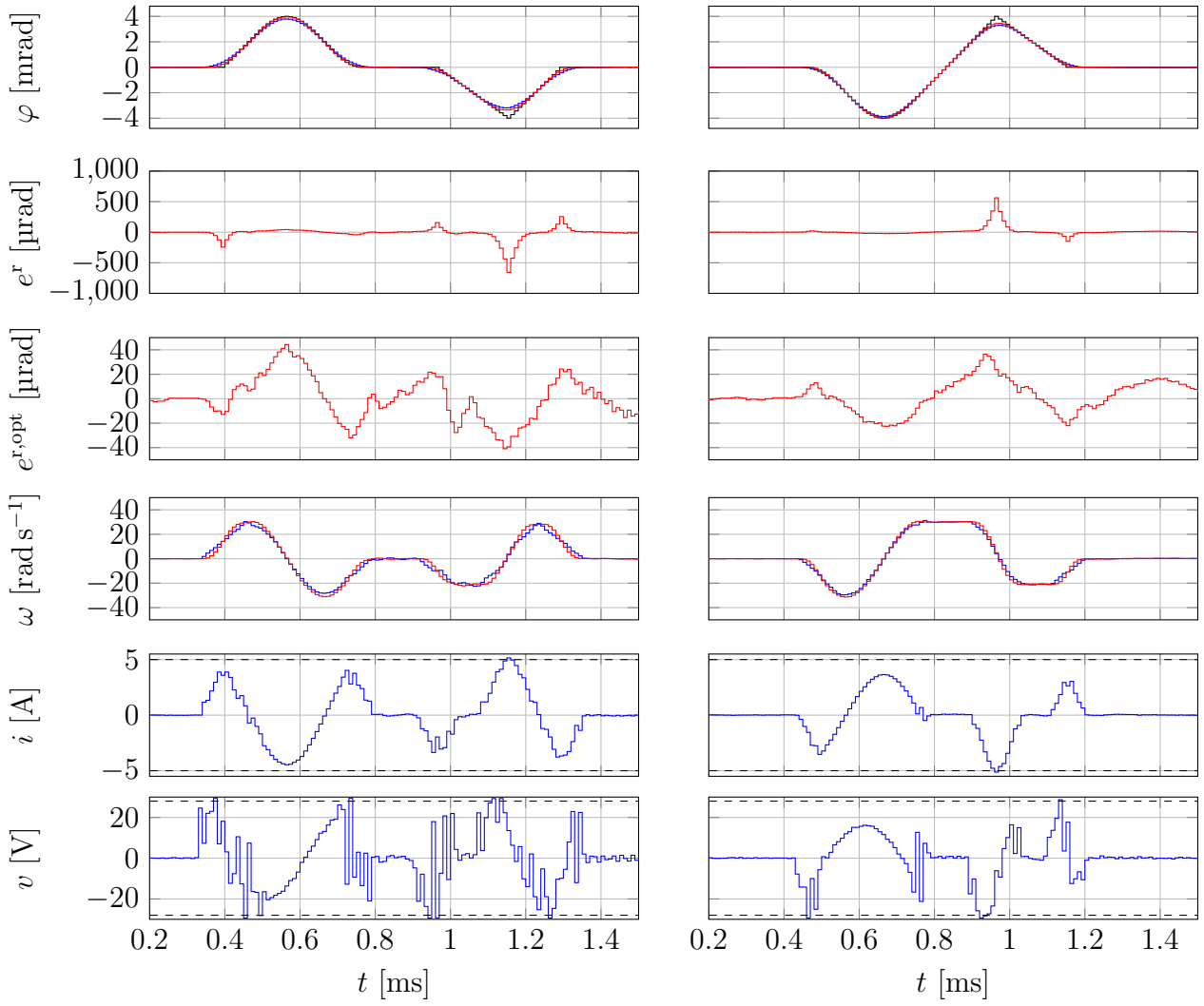
The results for the standard PRG and the near minimum-time PRG are shown in Figures 7.9 and 7.10 on pages 138/139. As both schemes respect the reduced current constraints of $\pm 5\,\text{A}$, the contour error is much higher compared with the advanced industry standard (Figure 7.8).

Figures 7.11 and 7.12 on pages 140/141 present the experimental results for the contouring PRG and the near minimum-time contouring PRG. Both approaches efficiently minimize the contour error by reducing the path speed in the curvatures *and* even respect the reduced current limit of $\pm 5\,\text{A}$. The magnitude of the contour error is even slightly smaller than the contour error of the advanced industry standard, which does *not* respect the reduced current limitations. The main difference between the contouring PRG and the near minimum-time contouring PRG is again the high-frequency content of the input voltage $v[k]$.

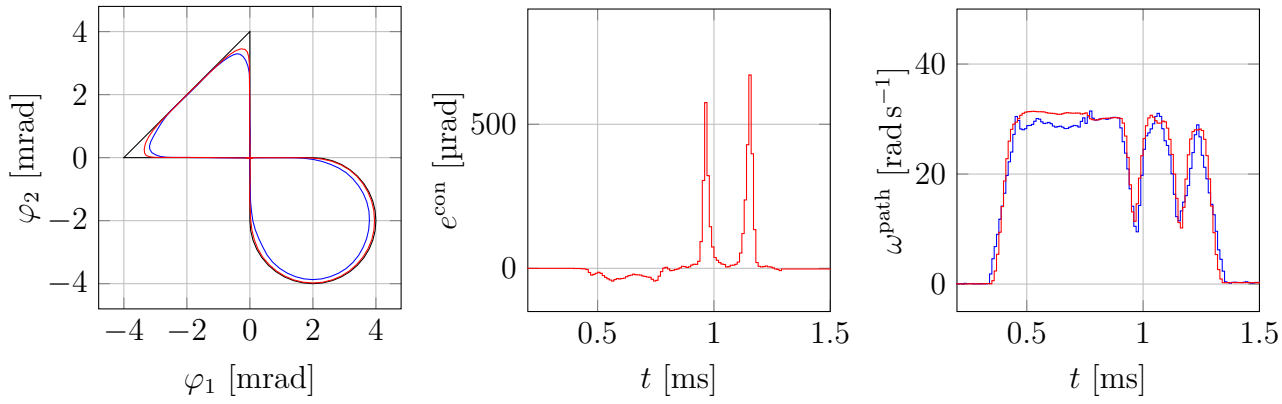
Summary

The experimental results show that the general PRG approach can inherently handle MIMO systems. The near minimum-time PRG leads again to reduced over- and undershoots compared with the standard PRG. Furthermore, the biaxial positioning example motivates PRG formulations that incorporate the contour error, which is the main performance measure in biaxial contouring, in the cost function. The measurement results confirm that the contour

error can be efficiently decreased by using PRGs that minimize the contour error compared with an advanced industry standard, the near minimum-time PRG, and the standard PRG. The contouring PRG and the near minimum-time contouring PRG lead to contour error evolutions that are only marginally above the noise level—the main difference is that the near minimum-time approach shows less high-frequency content. The evaluated reference paths show once more that the PRG approaches can efficiently handle arbitrary varying reference signals. The use of $N_{\text{shift}} > 1$ illustrates the flexibility of the general PRG scheme—the control system can be operated at a higher sampling rate than the respective PRG, or if the PRG is operated offline, the overall calculation time can be reduced.

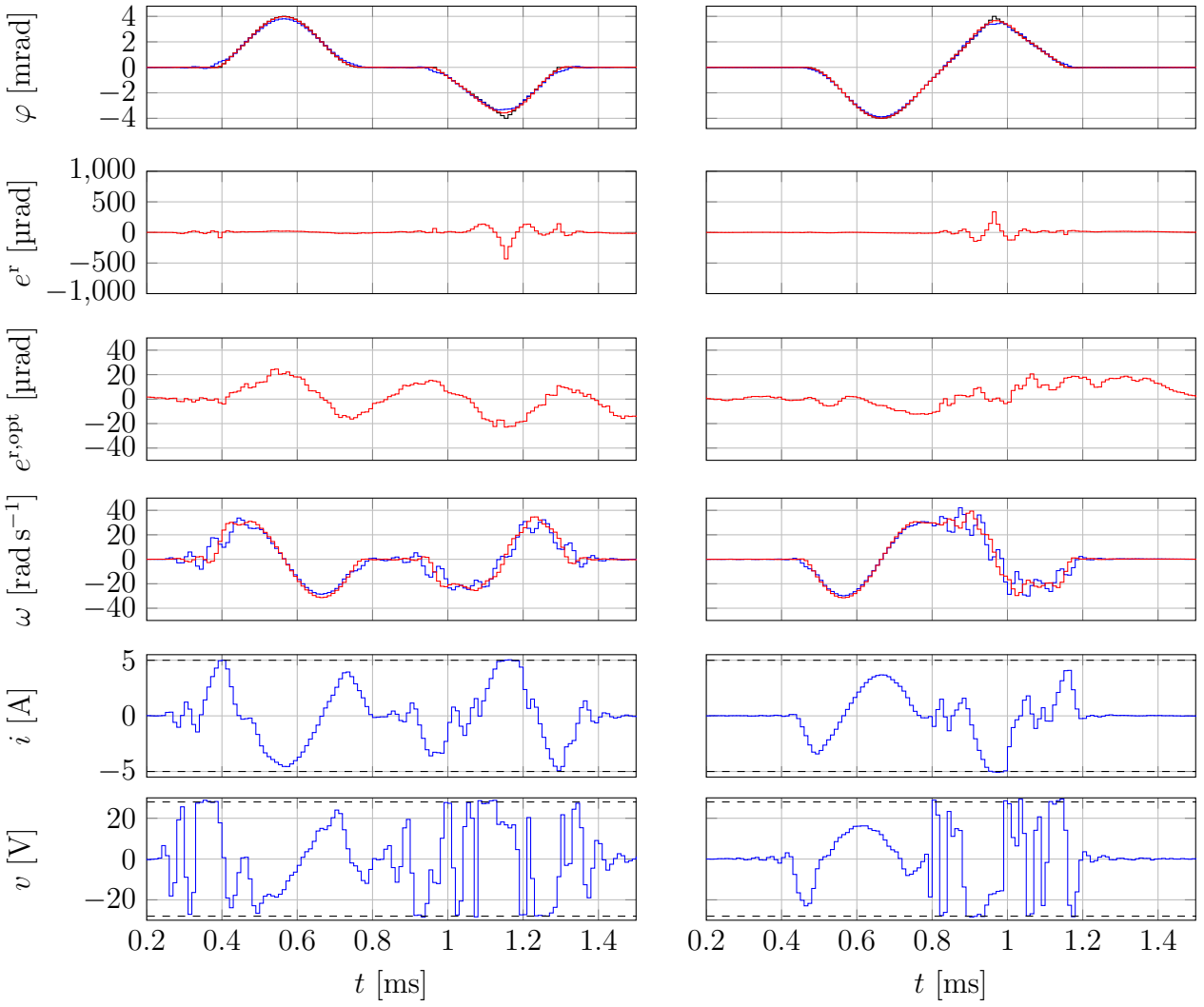


(a) Axis 1 (b) Axis 2

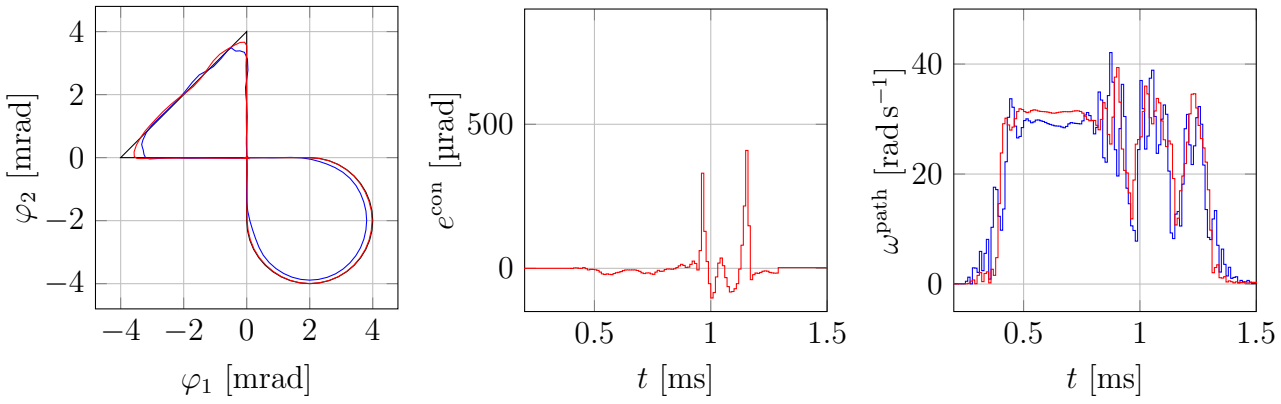


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.3: Biaxial contouring for a path with sharp corners: Advanced industry standard (see page 126 for the description of the signals)

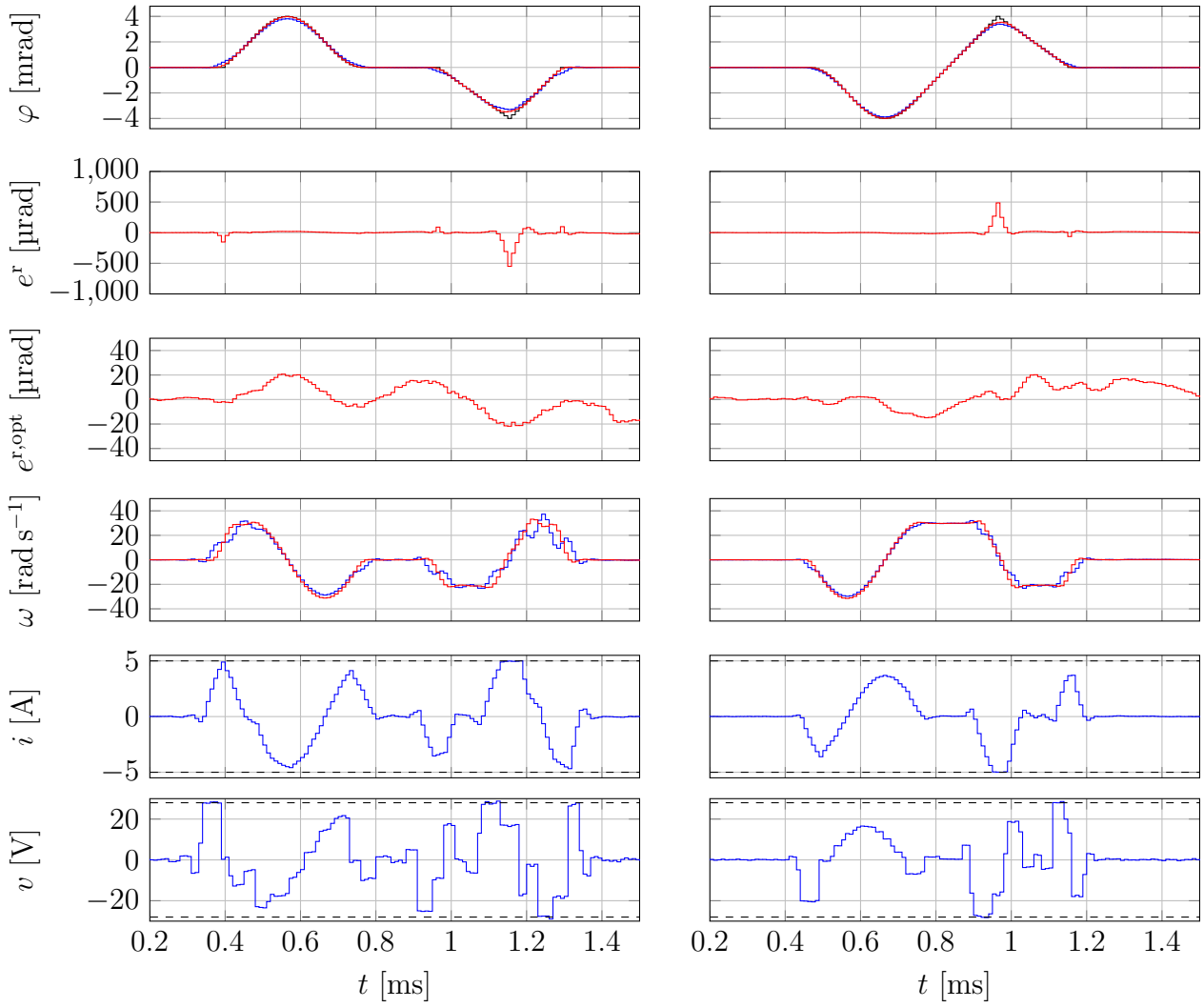


(a) Axis 1 (b) Axis 2

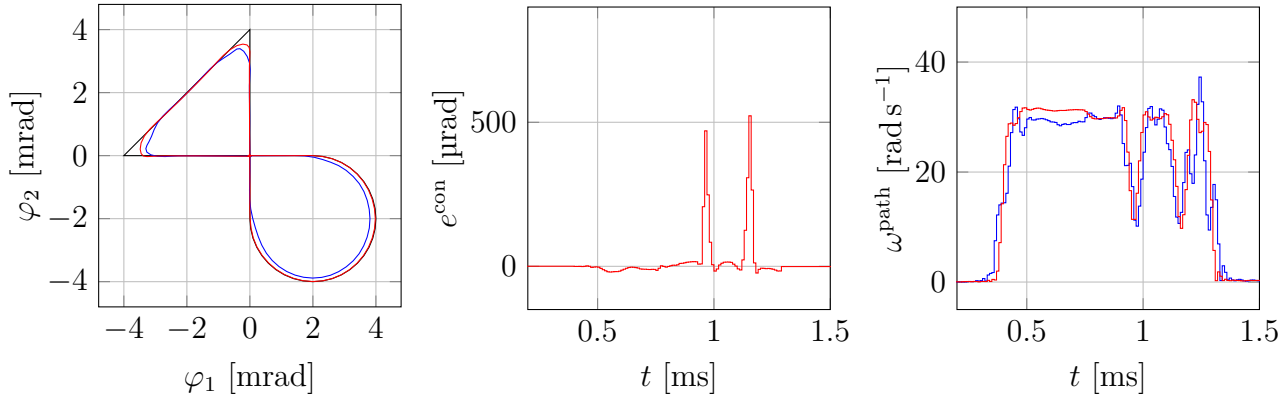


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.4: Biaxial contouring for a path with sharp corners: Standard PRG (see page 126 for the description of the signals)

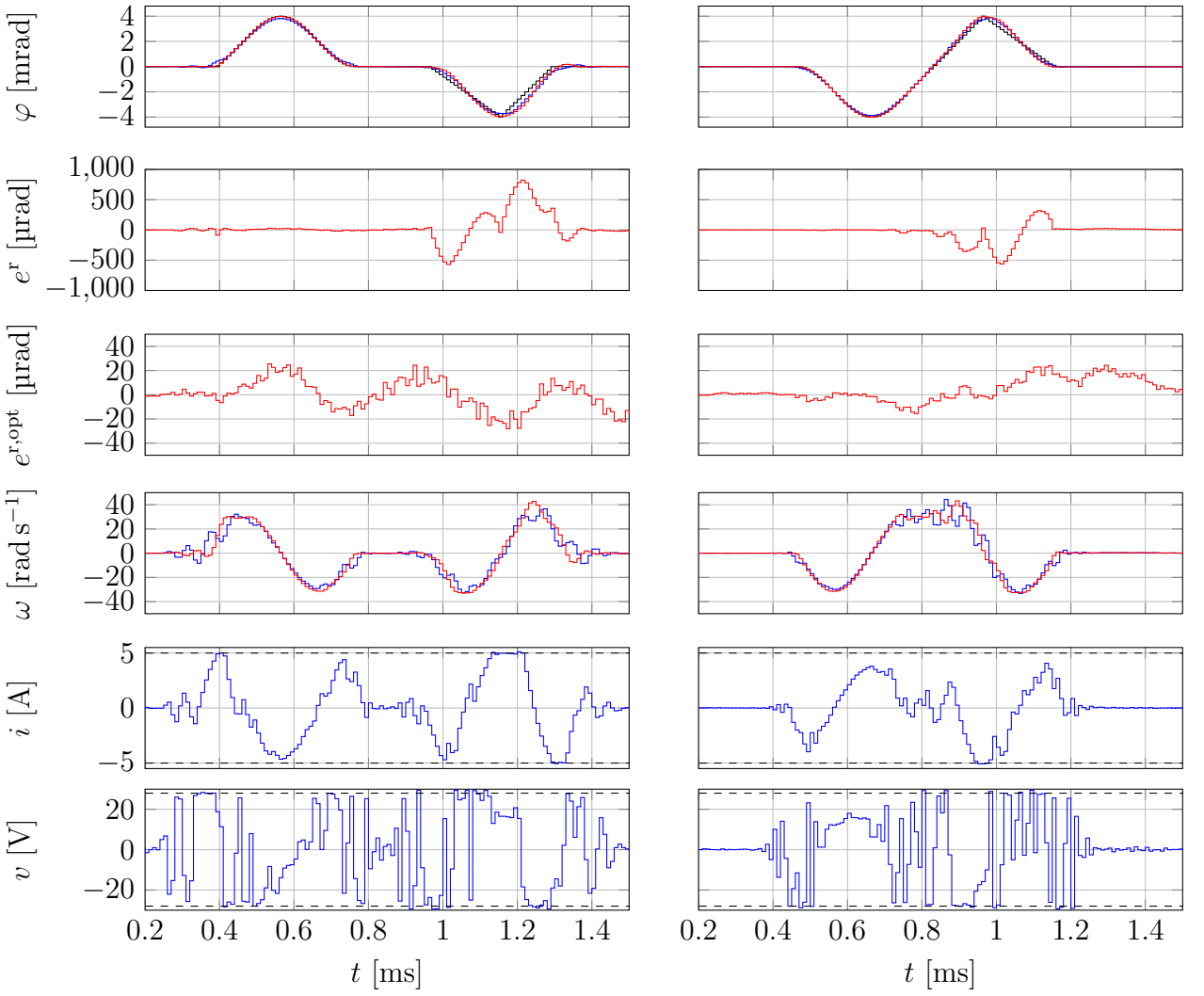


(a) Axis 1 (b) Axis 2

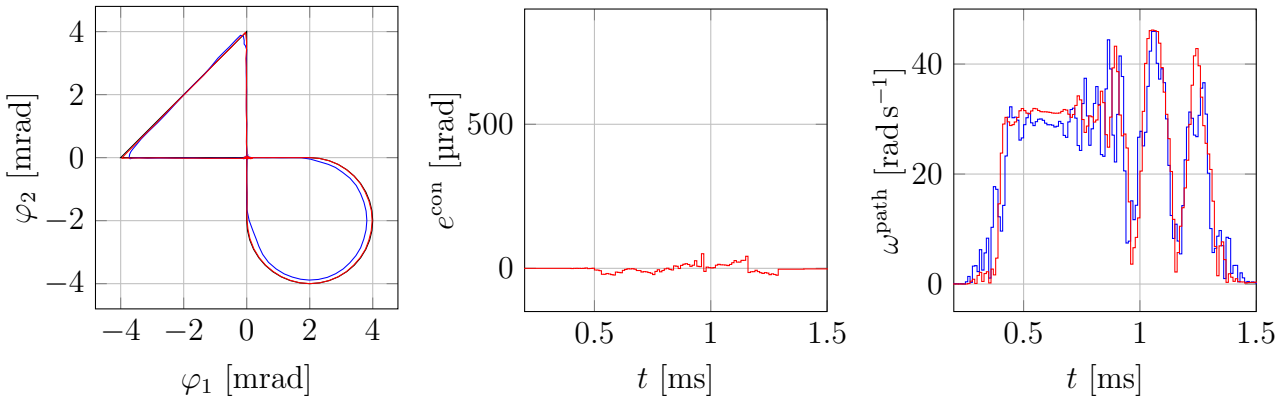


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.5: Biaxial contouring for a path with sharp corners: Near minimum-time PRG (see page 126 for the description of the signals)

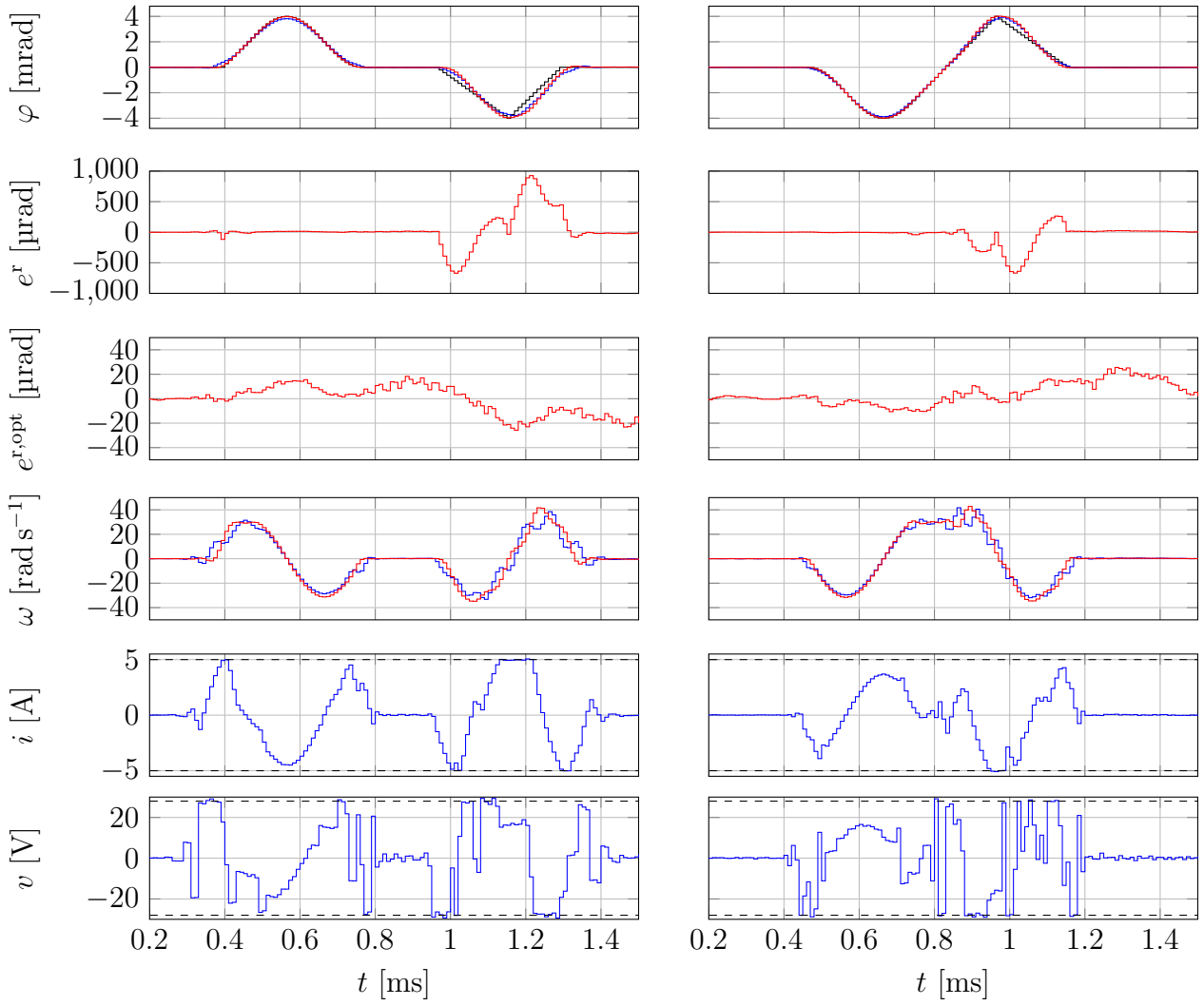


(a) Axis 1 (b) Axis 2

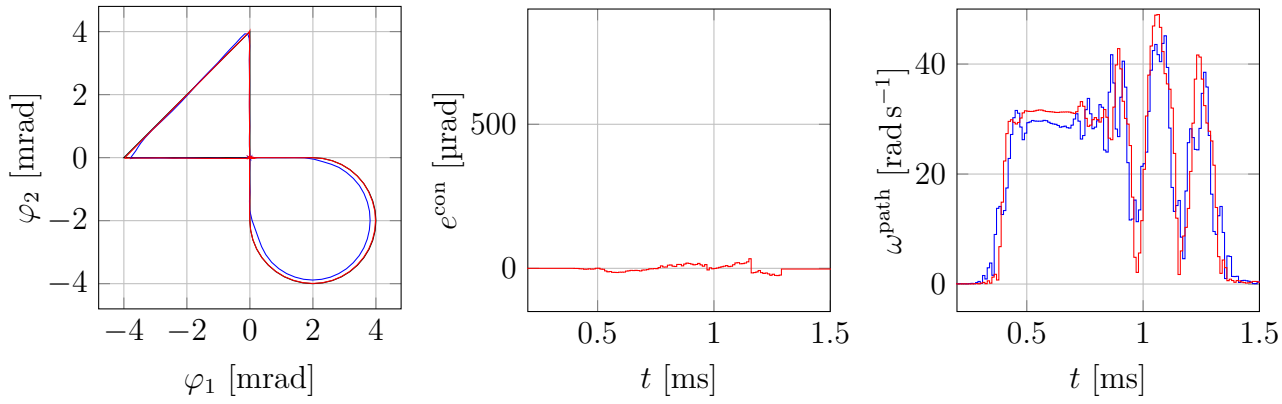


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.6: Biaxial contouring for a path with sharp corners: Contouring PRG (see page 126 for the description of the signals)

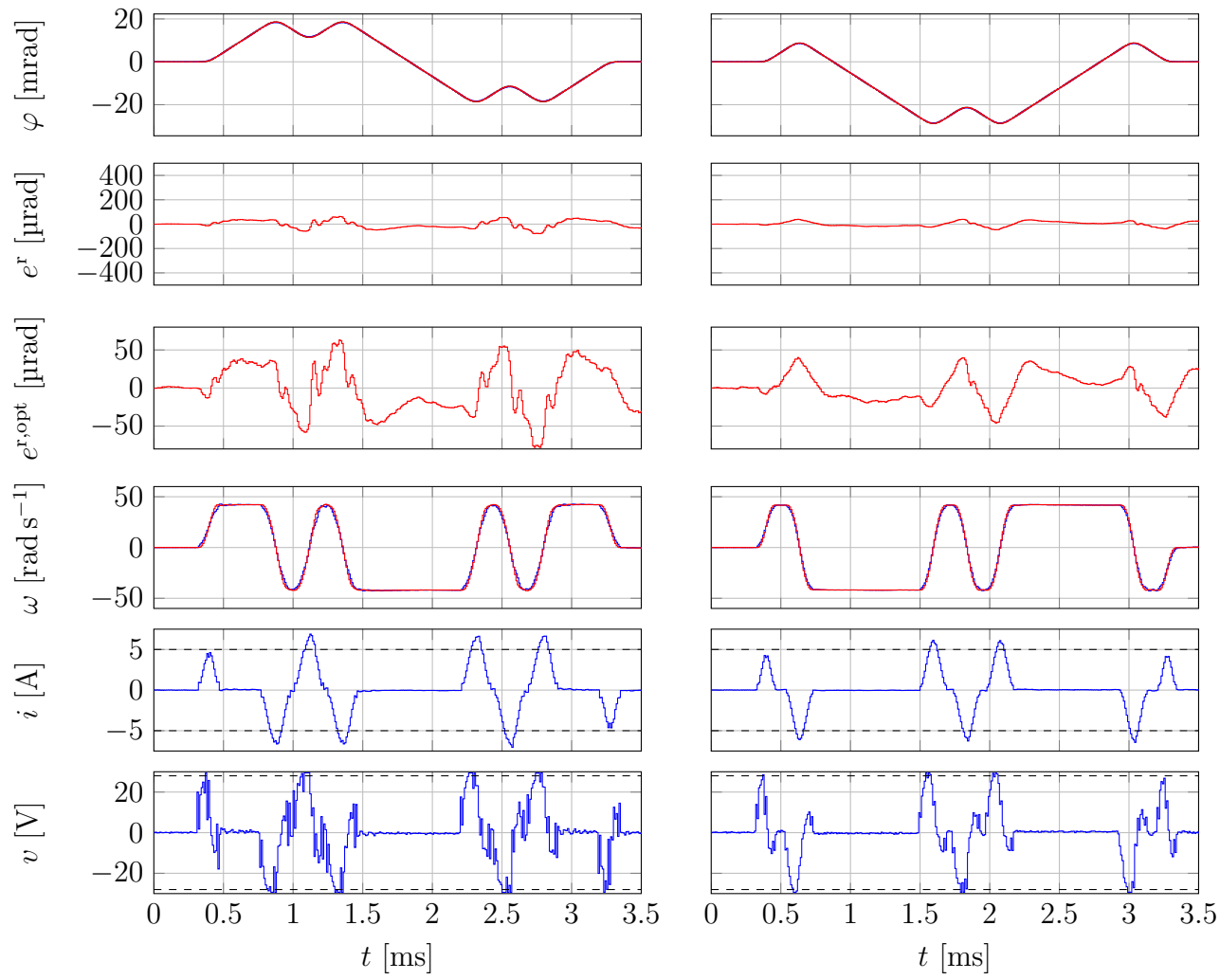


(a) Axis 1 (b) Axis 2

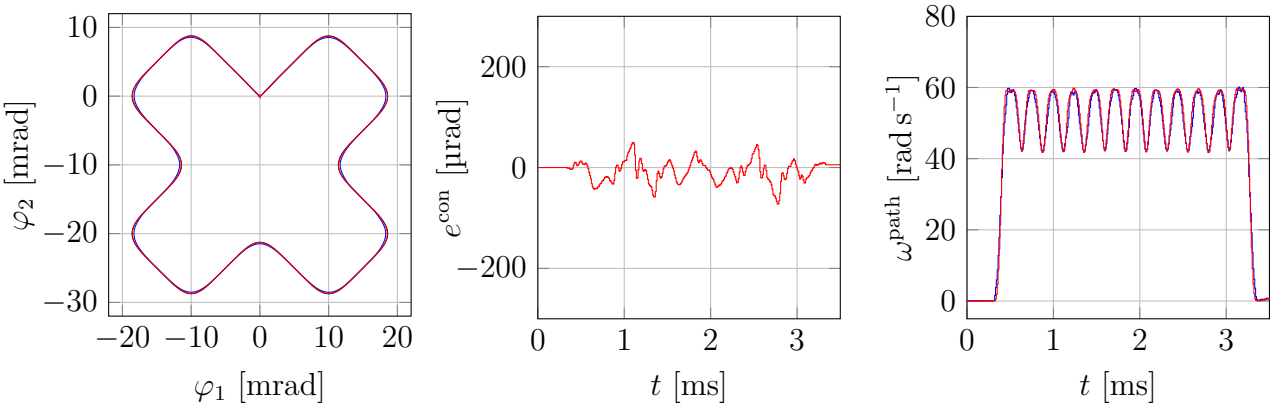


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.7: Biaxial contouring for a path with sharp corners: Near minimum-time contouring PRG (see page 126 for the description of the signals)

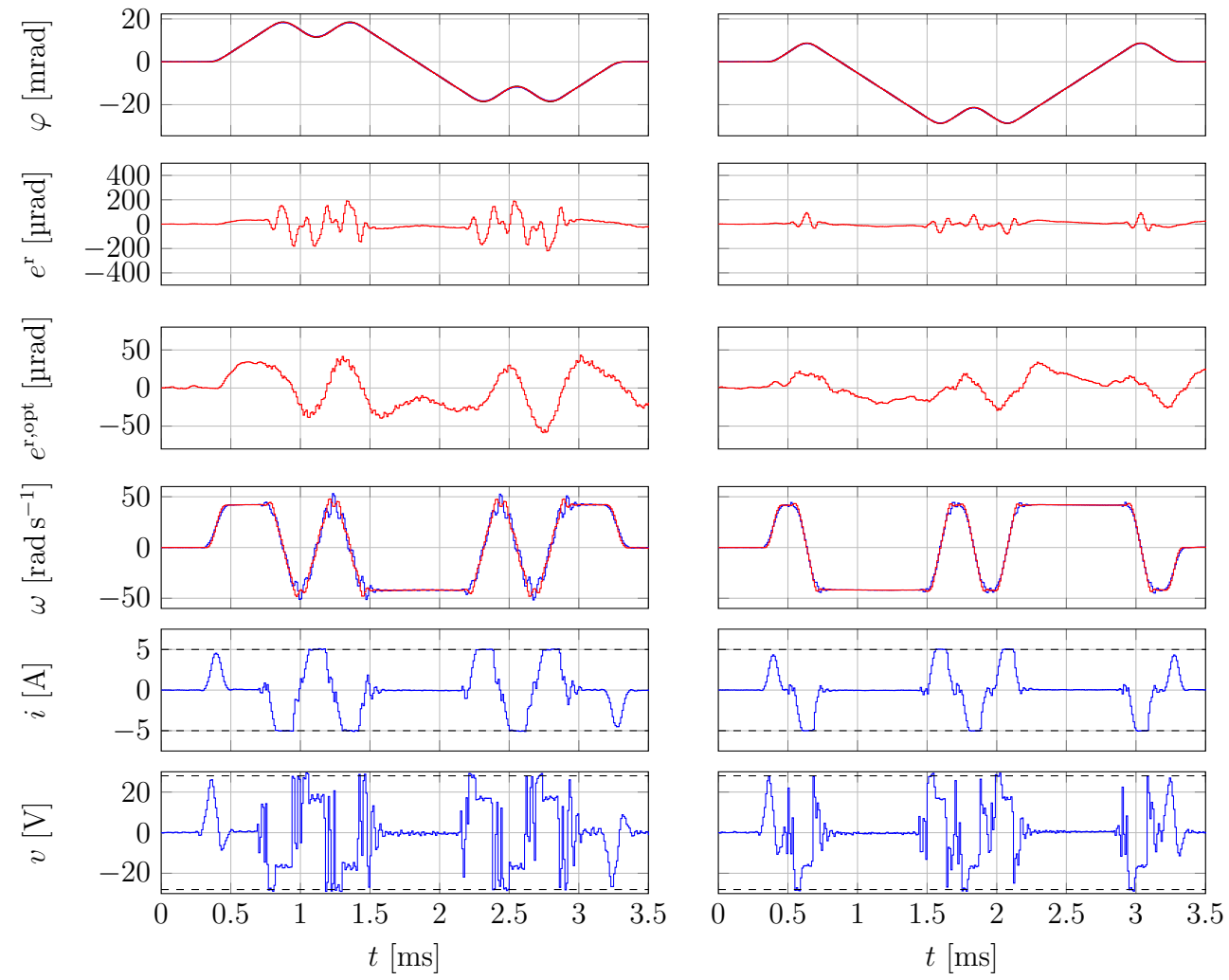


(a) Axis 1 (b) Axis 2

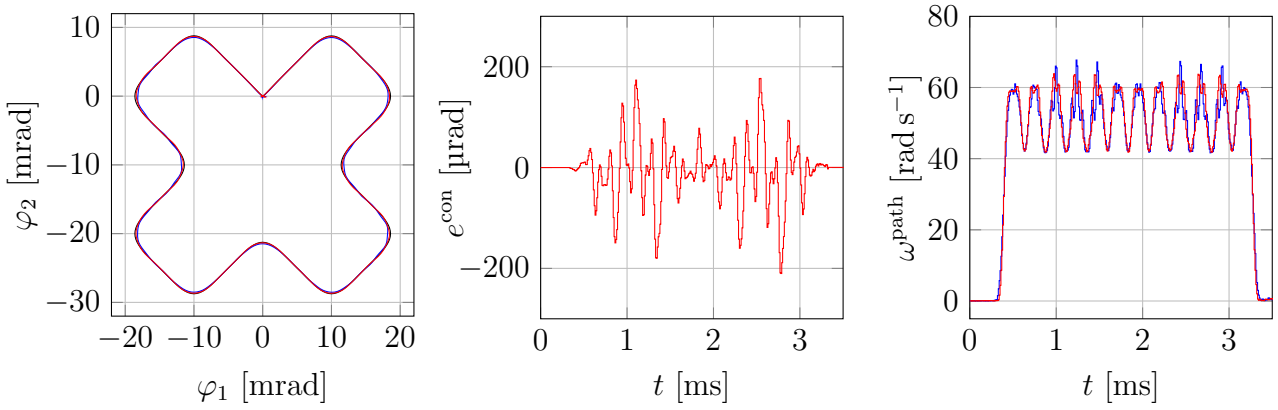


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.8: Biaxial contouring for a preprocessed path: Advanced industry standard (see page 126 for the description of the signals)

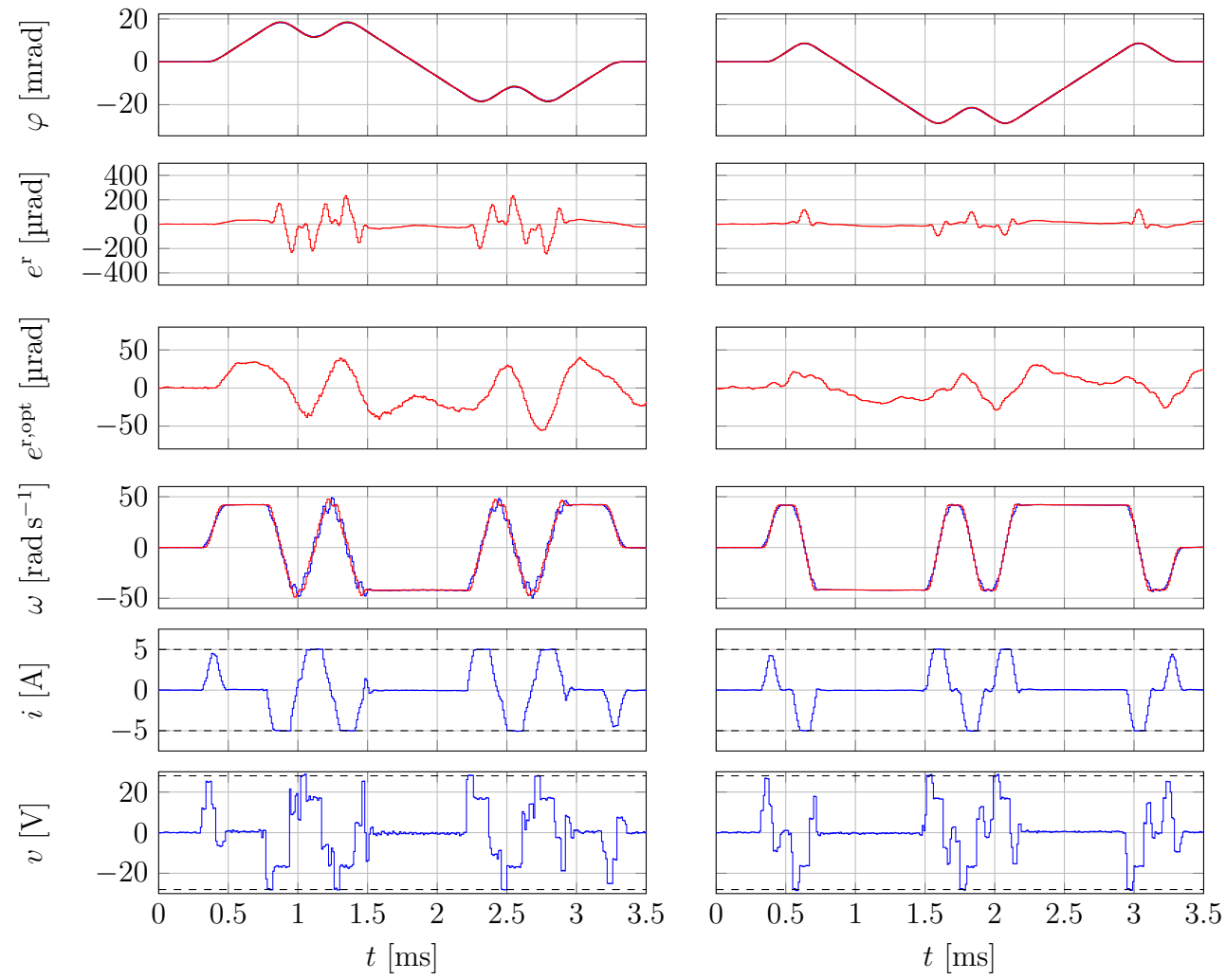


(a) Axis 1 (b) Axis 2

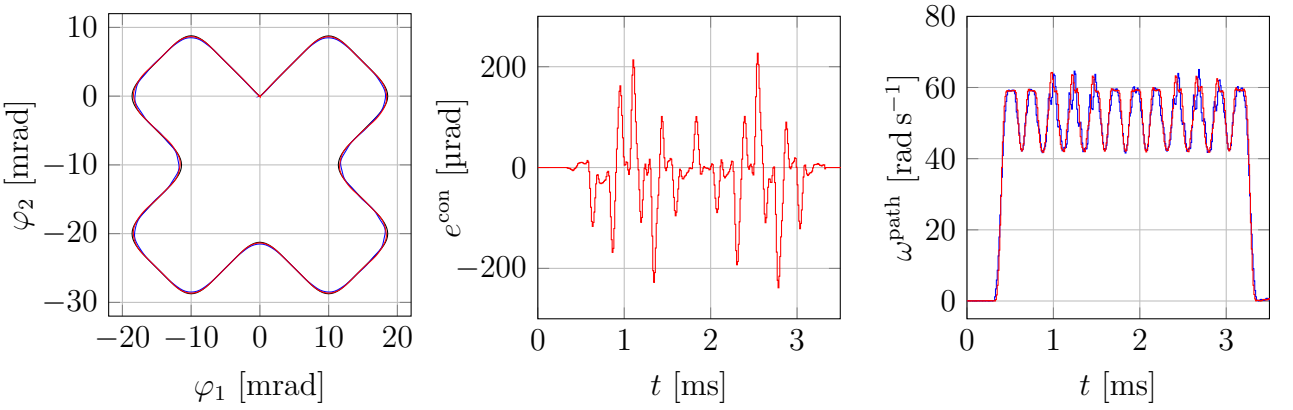


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.9: Biaxial contouring for a preprocessed path: Standard PRG (see page 126 for the description of the signals)

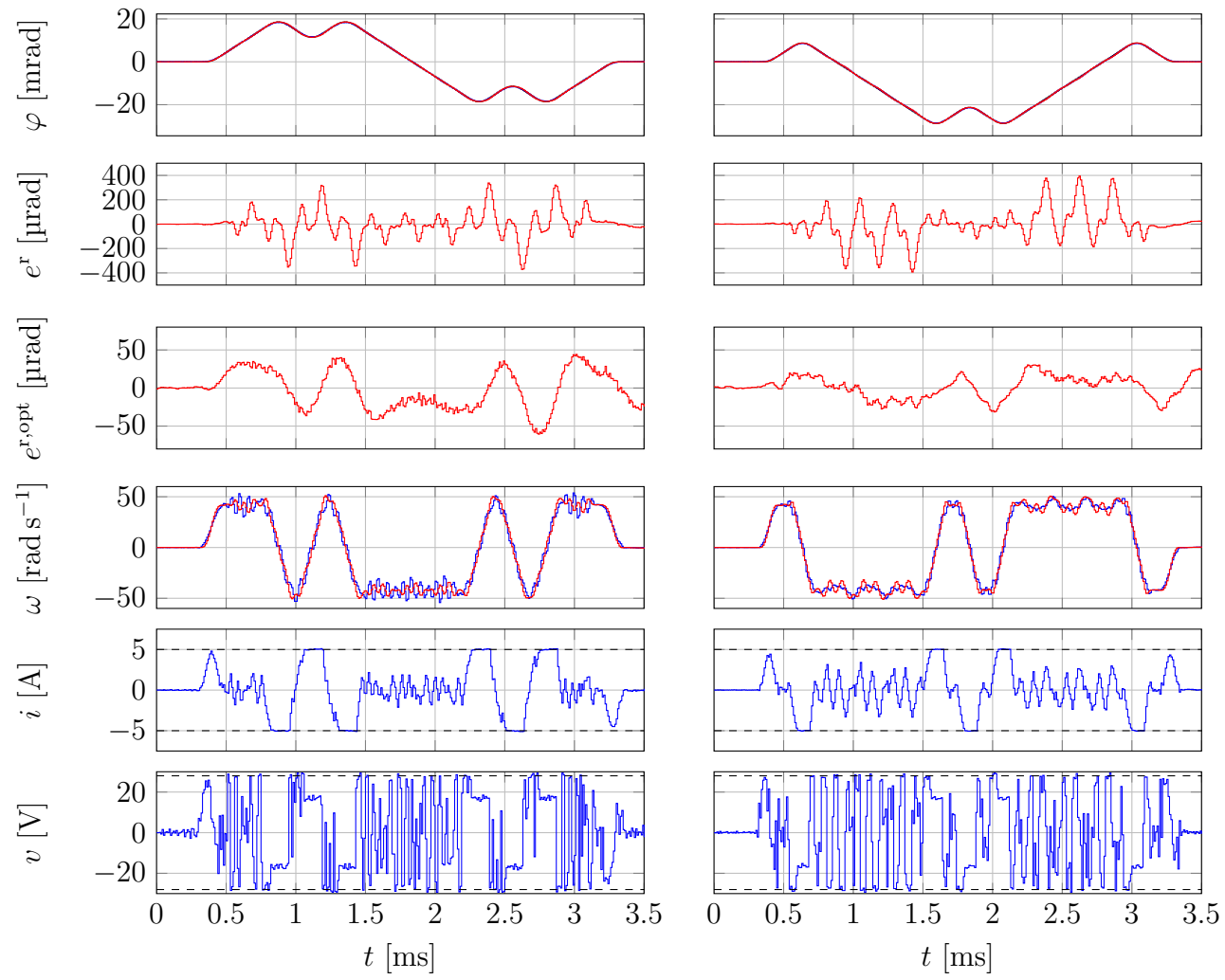


(a) Axis 1 (b) Axis 2

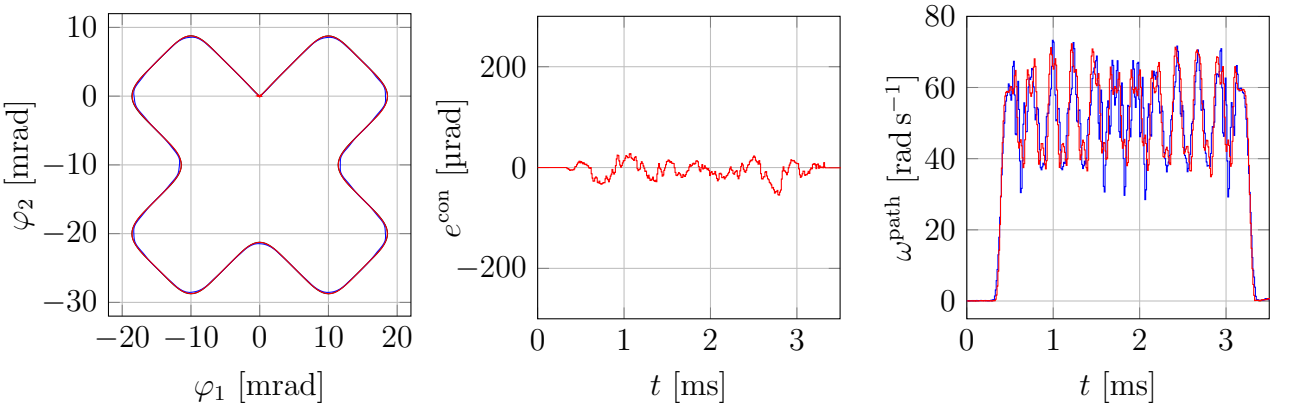


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.10: Biaxial contouring for a preprocessed path: Near minimum-time PRG (see page 126 for the description of the signals)

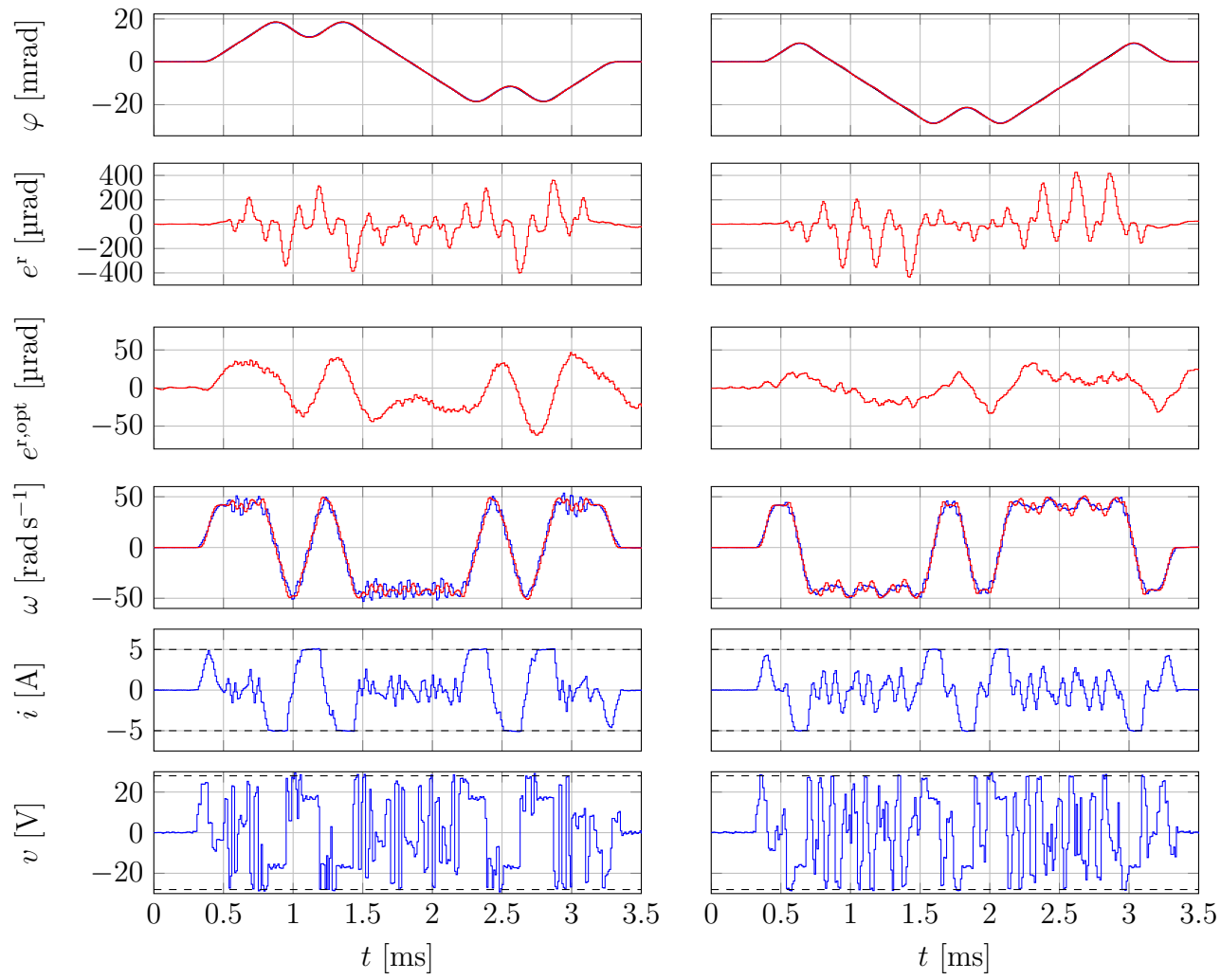


(a) Axis 1 (b) Axis 2

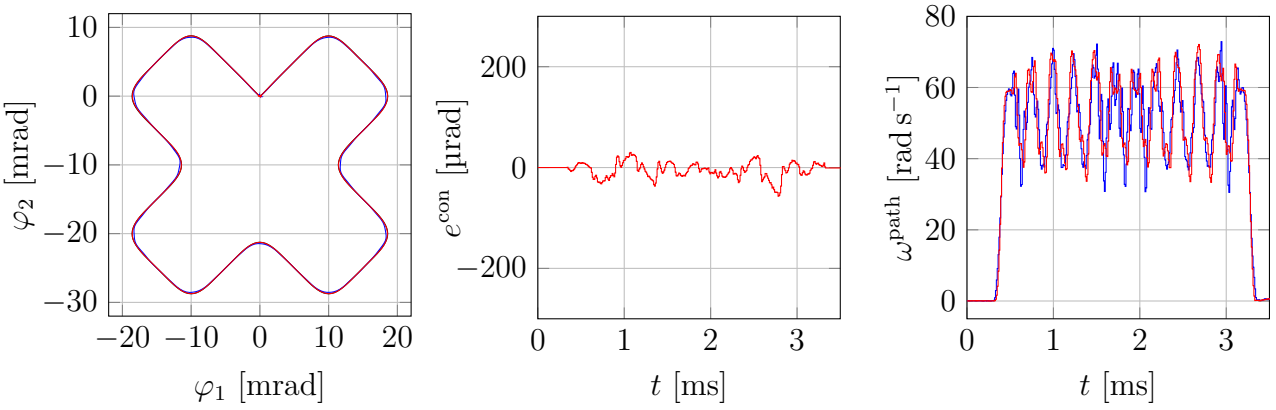


(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.11: Biaxial contouring for a preprocessed path: Contouring PRG (see page 126 for the description of the signals)



(a) Axis 1 (b) Axis 2



(c) Biaxial path (d) Contour error (e) Path speed

Figure 7.12: Biaxial contouring for a preprocessed path: Near minimum-time contouring PRG (see page 126 for the description of the signals)

8 Conclusions and Outlook

A predictive reference governor (PRG) scheme is proposed in this work in order to approach the physical limits of a constrained control system. A PRG is an add-on to a probably existing feedback controller and adapts the reference signal such that constraints on the control input and states are respected. Furthermore, the constraints should not only be respected but also exploited in order to achieve a fast transient response. The PRG concept is strongly related to model predictive control (MPC): It is based on the minimization of a cost function, inherently handles constraints, and is well suited for multiple-input, multiple-output (MIMO) systems. The involved optimization problem, however, is computationally intensive and is therefore hard to solve in real-time, especially for fast sampling control systems. To overcome this drawback, a multi-rate PRG scheme is introduced in Chapter 3, which allows to scale the sampling times to allow a real-time execution. In the extreme case, the PRG can be operated offline.

It is shown in Chapter 5 that a PRG that is based on a quadratic form, which is considered de facto standard, can lead to an overshoot in the step response. A PRG that is based on the ℓ_1-norm (sum of absolute values) is proposed in order to reduce this overshoot. Experimental results for a highly dynamic positioning system with a sampling time of $T_\mathrm{s} = 10\,\mu\mathrm{s}$ show that the ℓ_1-norm based approach can efficiently reduce this overshoot in a step response. Furthermore, compared with minimum-time optimal control, which represents the fastest transient response that is physically possibly, an ℓ_1-norm based PRG is able to achieve control results that are close to the minimum-time solution. Due to the low sampling time, however, all PRG approaches of this work were executed offline and therefore without feedback.

The concepts of robust model predictive control (RMPC) are used in Chapter 6 to design PRGs that offer an increased robustness against uncertain parameters. It is demonstrated experimentally that the ideas of RMPC can be used for the design of PRGs—even when they are operated offline and without feedback.

The flexibility of PRGs concerning the design of the cost function and the inherent handling of MIMO systems is emphasized in Chapter 7. An industrial biaxial positioning system, consisting of two electric motors, serves as an exemplary application. The positioning system is modeled as a MIMO system, which is used to design the PRGs. Furthermore, it is shown

that the contour error, which is the main performance measure in biaxial positioning, can be incorporated in the cost function. The experimental results demonstrate that the PRGs can efficiently decrease the contour error.

The PRG approaches of this thesis were executed offline because of the fast sampling control system that is needed for the considered positioning system and because of a comparably high prediction horizon. Therefore, an essential future step will be the real-time implementation of the PRG approaches. As the real-time operation allows using state feedback, a thorough analysis of the influence of the state feedback on the control results is necessary, especially concerning measurement noise.

A future research direction could be the use of PRGs for fault detection. The experimental results for the considered positioning system show that nearly perfect tracking of the adapted/optimized reference trajectories is achieved by the underlying controller. The optimized control error, which is the difference between the adapted/optimized reference and the system output, is a measure for the tracking behavior. It is shown in this work that an optimized control error that is slightly higher than the noise level can be achieved through the handling of constraints and the use of a precise system model. Thus, it should be possible to detect faults by analyzing the evolution of the optimized control error.

Summing up, this thesis shows that the concept of PRGs enables to exploit the features of MPC even for fast sampling control systems, which do not yet allow real-time operation. Furthermore, by designing a PRG that relies on an ℓ_1-norm based cost function, near minimum-time control results can be achieved.

A Predictive Reference Governor Algorithms

Algorithm 1 Predictive reference governor (PRG)

initialize the system model
$k = 0$
$\boldsymbol{x}_{\text{pred}}[0] = \boldsymbol{x}_0$ (initial state)
set the prediction horizon N
determine the cost function weights through tuning
calculate the matrices/vectors in the optimization problem (3.1) that do *not* vary over time
is feedback available? ($\text{feedback}_{\text{PRG}} = \text{true}$ or $\text{feedback}_{\text{PRG}} = \text{false}$)
should the PRG work online? ($\text{online}_{\text{PRG}} = \text{true}$ or $\text{online}_{\text{PRG}} = \text{false}$)

if $\text{online}_{\text{PRG}}$ **then**
    determine n_{mr} such that the main loop of this algorithm can be executed at $n_{\text{mr}} \cdot T_{\text{s}}$

    if $n_{\text{mr}} > N$ **then**
        online execution of the main loop is not possible
        $\text{online}_{\text{PRG}} = \text{false}$
    else
        $N_{\text{shift}} = n_{\text{mr}}$
        initialize $\text{interrupt}_{\text{PRG}}$ at $n_{\text{mr}} \cdot T_{\text{s}}$
    end if
end if

if not $\text{online}_{\text{PRG}}$ **then**
    $\text{feedback}_{\text{PRG}} = \text{false}$
    set N_{shift}, where $N_{\text{shift}} \leq N$ must hold
end if

main loop: see Algorithm 2

if not $\text{online}_{\text{PRG}}$ **then**
    copy $\mathcal{U}^{\text{r,opt}}[k]$ and $\mathcal{X}^{\text{r,opt}}[k]$ to the controller reference buffer
    start the controller
end if

Algorithm 2 Main loop predictive reference governor (PRG)

while reference signal available **do**
  if $\text{online}_{\text{PRG}}$ **then**
    while not $\text{interrupt}_{\text{PRG}}$ **do**
      wait
    end while
  end if

  if $\text{feedback}_{\text{PRG}}$ **then**
    $\boldsymbol{x}[k]$ is determined through measurement or estimation
  else
    $\boldsymbol{x}[k] = \boldsymbol{x}_{\text{pred}}$ is determined by prediction/simulation in the previous iteration
  end if

  read the reference signal $\boldsymbol{Y}^{\text{r}}[k]$
  determine the optimization problem in standard form (3.1) and solve it
  the optimal sequence of inputs is $\boldsymbol{U}^{\star}[k] = \left[\, \boldsymbol{u}^{\star}[k]^{\top} \; \ldots \; \boldsymbol{u}^{\star}[k+N_{\text{shift}}-1]^{\top} \; \ldots \; \boldsymbol{u}^{\star}[k+N-1]^{\top} \,\right]^{\top}$

  predict the state trajectories $\boldsymbol{X}[k] = \mathcal{A}_x \boldsymbol{x}[k] + \mathcal{B}_x \boldsymbol{U}^{\star}[k]$ (see (2.15))
  where $\boldsymbol{X}[k] = \left[\, \boldsymbol{x}[k+1]^{\top} \; \ldots \; \boldsymbol{x}[k+N_{\text{shift}}-1]^{\top} \; \boldsymbol{x}[k+N_{\text{shift}}]^{\top} \; \ldots \; \boldsymbol{x}[k+N]^{\top} \,\right]^{\top}$
  $\boldsymbol{x}_{\text{pred}} = \boldsymbol{x}[k + N_{\text{shift}}]$

  $\boldsymbol{U}^{\star}_{\text{shift}}[k] = \left[\, \boldsymbol{u}^{\star}[k]^{\top} \; \ldots \; \boldsymbol{u}^{\star}[k+N_{\text{shift}}-1]^{\top} \,\right]^{\top}$
  $\boldsymbol{X}_{\text{shift}}[k] = \left[\, \boldsymbol{x}[k]^{\top} \; \ldots \; \boldsymbol{x}[k+N_{\text{shift}}-1]^{\top} \,\right]^{\top}$

  if $\text{online}_{\text{PRG}}$ **then**
    $\mathcal{U}^{\text{r,opt}}[k] = \boldsymbol{U}^{\star}_{\text{shift}}[k]$
    $\mathcal{X}^{\text{r,opt}}[k] = \boldsymbol{X}_{\text{shift}}[k]$
    copy $\mathcal{U}^{\text{r,opt}}[k]$ and $\mathcal{X}^{\text{r,opt}}[k]$ to the buffer of the rate transition
  else
    if $k == 0$ **then**
      $\mathcal{U}^{\text{r,opt}}[k] = \boldsymbol{U}^{\star}_{\text{shift}}[k]$
      $\mathcal{X}^{\text{r,opt}}[k] = \boldsymbol{X}_{\text{shift}}[k]$
    else
      append: $\mathcal{U}^{\text{r,opt}}[k] = \left[\, \mathcal{U}^{\text{r,opt}}[k-N_{\text{shift}}]^{\top} \; \boldsymbol{U}^{\star}_{\text{shift}}[k] \,\right]^{\top}$
      append: $\mathcal{X}^{\text{r,opt}}[k] = \left[\, \mathcal{X}^{\text{r,opt}}[k-N_{\text{shift}}]^{\top} \; \boldsymbol{X}_{\text{shift}}[k] \,\right]^{\top}$
    end if
  end if

  $k = k + N_{\text{shift}}$
end while

Nomenclature

$\boldsymbol{A}^\top$	Transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{-1}$	Inverse of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{-\top}$	Transpose and inverse of matrix $\boldsymbol{A}$
$\boldsymbol{A} \succ (\succeq) 0$	Positive (semi-)definite matrix
$\boldsymbol{A} \prec (\preceq) 0$	Negative (semi-)definite matrix
$\mathrm{Co}\,\{\}$	Convex hull
$\det(\boldsymbol{A})$	Determinant of matrix $\boldsymbol{A}$
$\mathrm{rank}\,(\boldsymbol{A})$	Rank of matrix $\boldsymbol{A}$
$\oplus_{k=1}^{N} \boldsymbol{A}_k$	Block diagonal matrix with matrices $\boldsymbol{A}_1$ to $\boldsymbol{A}_N$ along its diagonal
$\mathrm{diag}\{\boldsymbol{a}\}$	Diagonal matrix with a vector $\boldsymbol{a}$ on its diagonal
$\boldsymbol{I}_x$	Identity matrix of dimension $x \times x$
$\mathbf{1}_{x\times y}$	One matrix of dimension $x \times y$
$\mathbf{0}_{x\times y}$	Zero matrix of dimension $x \times y$
$\lvert\boldsymbol{a}\rvert$	Element-wise absolute value of vector $\boldsymbol{a}$
$\lVert\boldsymbol{a}\rVert$ or $\lVert\boldsymbol{a}\rVert_2$	Euclidean norm of vector $\lVert\boldsymbol{a}\rVert_2 = \sqrt{\boldsymbol{a}^\top\boldsymbol{a}} = \sqrt{\sum_{i=1}^{n_a} a_i^2}$
$\lVert\boldsymbol{a}\rVert_2^2$	Quadratic form of vector $\lVert\boldsymbol{a}\rVert_2^2 = \boldsymbol{a}^\top\boldsymbol{a} = \sum_{i=1}^{n_a} a_i^2$
$\lVert\boldsymbol{a}\rVert_1$	ℓ_1-norm of vector $\lVert\boldsymbol{a}\rVert_1 = \sum_{i=1}^{n_a} \lvert a_i\rvert$

Symbols

$\boldsymbol{A}$	Discrete state matrix (gained through ZOH transformation of the respective continuous matrix $\boldsymbol{A}_\mathrm{c}$). For the SISO case only one axis is considered ($\boldsymbol{A} = \boldsymbol{A}_1$ or $\boldsymbol{A} = \boldsymbol{A}_2$); in the MIMO case two axes are combined ($\boldsymbol{A} = \left[\begin{smallmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2 \end{smallmatrix}\right]$)
$\mathcal{A}_x$	State prediction matrix for a prediction in the form of $\boldsymbol{X}[k] = \mathcal{A}_x \boldsymbol{x}[k] + \mathcal{B}_x \boldsymbol{U}[k]$
$\mathcal{A}_y$	Output prediction matrix for a prediction in the form of $\boldsymbol{Y}[k] = \mathcal{A}_y \boldsymbol{x}[k] + \mathcal{B}_y \boldsymbol{U}[k]$
$\boldsymbol{B}$	Discrete input matrix (gained through ZOH transformation of the respective continuous matrix $\boldsymbol{B}_\mathrm{c}$). For the SISO case only one axis is considered ($\boldsymbol{B} = \boldsymbol{B}_1$ or $\boldsymbol{B} = \boldsymbol{B}_2$); in the MIMO case two axes are combined ($\boldsymbol{B} = \left[\begin{smallmatrix} \boldsymbol{B}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_2 \end{smallmatrix}\right]$)
$\mathcal{B}_\Delta$	Rate of input change transformation matrix ($\boldsymbol{\Delta U}[k] = \mathcal{B}_\Delta \boldsymbol{U}[k]$)
$\boldsymbol{\beta}$	Auxiliary variable introduced by the transformation of an ℓ_1-norm based optimization problem to an LP
$\mathcal{B}_x$	State prediction matrix for a prediction in the form of $\boldsymbol{X}[k] = \mathcal{A}_x \boldsymbol{x}[k] + \mathcal{B}_x \boldsymbol{U}[k]$
$\mathcal{B}_y$	Output prediction matrix for a prediction in the form of $\boldsymbol{Y}[k] = \mathcal{A}_y \boldsymbol{x}[k] + \mathcal{B}_y \boldsymbol{U}[k]$
$\boldsymbol{C}$	Discrete and continuous output matrix. For the SISO case only one axis is considered ($\boldsymbol{C} = \boldsymbol{C}_1$ or $\boldsymbol{C} = \boldsymbol{C}_2$); in the MIMO case two axes are combined ($\boldsymbol{C} = \left[\begin{smallmatrix} \boldsymbol{C}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2 \end{smallmatrix}\right]$)
c	Two-mass system (motor and load) spring constant
C^ISE	ISE criterion
C^ROB	Robust performance criterion
$\boldsymbol{D}$	Discrete and continuous feedthrough matrix
d	Two-mass system (motor and load) damping constant

Δe^{r}	Difference of the control errors of two uncertainty realizations
$\boldsymbol{\Delta U}$	Stacked vector of the rate of input change
$\boldsymbol{E}^{\mathrm{r}}$	Stacked control error vector (predicted)
e^{con}	Contour error of a biaxial positioning system
$\tilde{\boldsymbol{E}}^{\mathrm{con}}$	Stacked approximated contour error vector (predicted) of a biaxial positioning system
$\tilde{e}^{\mathrm{con}}$	Approximated contour error of a biaxial positioning system
e^{est}	Estimation error. Angular motor position estimation errors: $e^{\mathrm{est}}_{\mathrm{m},1}$, $e^{\mathrm{est}}_{\mathrm{m},2}$; angular load position estimation errors: $e^{\mathrm{est}}_{\mathrm{l},1}$, $e^{\mathrm{est}}_{\mathrm{l},2}$
e^{lag}	Lag distance of a biaxial positioning system
$\tilde{\boldsymbol{E}}^{\mathrm{lag}}$	Stacked approximated lag distance vector (predicted) of a biaxial positioning system
$\tilde{e}^{\mathrm{lag}}$	Approximated lag distance of a biaxial positioning system
e^{s}	State error used for DynFF
e^{r}	Control error (difference between the *unoptimized* reference and the system output)
$e^{\mathrm{r,opt}}$	Optimized control error (difference between the *optimized* reference and the system output)
$\boldsymbol{F}_{\ell_1}$	Auxiliary variable in an ℓ_1-norm based cost function in the form of $\|\boldsymbol{F}_{\ell_1}[k]\boldsymbol{U}[k] + \boldsymbol{g}_{\ell_1}[k]\|_1$
$\boldsymbol{F}_{\mathrm{q}}$	Auxiliary variable in a quadratic form based cost function in the form of $\|\boldsymbol{F}_{\mathrm{q}}[k]\boldsymbol{U}[k] + \boldsymbol{g}_{\mathrm{q}}[k]\|_2^2$
f	Frequency
f_{PWM}	Sampling frequency of the PWM stage
$\mathcal{G}$	Constraint matrix for element-wise constraints in the form of $\mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]$
$\boldsymbol{g}_{\ell_1}$	Auxiliary variable in an ℓ_1-norm based cost function in the form of $\|\boldsymbol{F}_{\ell_1}[k]\boldsymbol{U}[k] + \boldsymbol{g}_{\ell_1}[k]\|_1$
$\boldsymbol{g}_{\mathrm{q}}$	Auxiliary variable in a quadratic form based cost function in the form of $\|\boldsymbol{F}_{\mathrm{q}}[k]\boldsymbol{U}[k] + \boldsymbol{g}_{\mathrm{q}}[k]\|_2^2$
$\mathcal{H}$	Stacked constraint vector for element-wise constraints in the form of $\mathcal{G}\boldsymbol{U}[k] \leq \mathcal{H}[k]$
$\boldsymbol{H}(z)$	Matrix of discrete-time transfer functions
θ	Angle of the tangent to a reference path in biaxial contouring

i	Motor current. The specific axes are denoted by i_1 and i_2.
i^{d}	Estimated disturbance current
$i_{\max}$	Maximum motor current
i^{meas}	Measured motor current. The specific axes are denoted by i_1^{meas} and i_2^{meas}.
J_{l}	Load moment of inertia
J_{m}	Motor moment of inertia
$J(\cdot)$	General cost function
$J_{\ell_1}(\cdot)$	Cost function based on the ℓ_1-norm
$J_{\ell_1,\mathrm{con}}(\cdot)$	Cost function based on the ℓ_1-norm (contour error formulation)
$J_{\ell_1,\mathrm{rob}}(\cdot)$	Cost function based on the ℓ_1-norm (robust formulation)
$J_{\mathrm{q}}(\cdot)$	Cost function based on a quadratic form
$J_{\mathrm{q,con}}(\cdot)$	Cost function based on a quadratic form (contour error formulation)
$J_{\mathrm{q,rob}}(\cdot)$	Cost function based on a quadratic form (robust formulation)
$\boldsymbol{K}$	State-space feedback control gain matrix used for pole-placement
k	Sampling instant
K_{f}	Motor viscous friction coefficient
K_{t}	Motor torque constant
L	Motor inductance
N	Prediction horizon
n_{mr}	Multi-rate factor (the slower part of a multi-rate scheme is operated with a sampling time of $n_{\mathrm{mr}} \cdot T_{\mathrm{s}}$)
n_{p}	Number of uncertain parameters
n_{seq}	Number of uncertain prediction sequences
N_{shift}	Receding horizon shift
n_{sw}	Number of switchings of an input constrained minimum-time optimal control problem for an LTI system
n_u	Number of system inputs
n_x	Number of system states
n_y	Number of system outputs
$\boldsymbol{Q}_{\ell_1}$	Control error weighting matrix $\boldsymbol{Q}_{\ell_1} = \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{q}_{\ell_1}\}$ depending on the weighting vector $\boldsymbol{q}_{\ell_1}$ (for an ℓ_1-norm based cost function)
$\boldsymbol{Q}_{\ell_1}^{\mathrm{con}}$	Biaxial contour error weighting matrix $\boldsymbol{Q}_{\ell_1}^{\mathrm{con}} = \oplus_{i=1}^{N} \mathrm{diag}\{q_{\ell_1}^{\mathrm{con}}\}$ depending on the weight $q_{\ell_1}^{\mathrm{con}}$ (for an ℓ_1-norm based cost function)

$\boldsymbol{Q}_{\ell_1}^{\mathrm{lag}}$	Biaxial lag distance weighting matrix $\boldsymbol{Q}_{\ell_1}^{\mathrm{lag}} = \oplus_{i=1}^{N} \mathrm{diag}\{q_{\ell_1}^{\mathrm{lag}}\}$ depending on the weight $q_{\ell_1}^{\mathrm{lag}}$ (for an ℓ_1-norm based cost function)
$\boldsymbol{Q}_{\mathrm{q}}$	Control error weighting matrix $\boldsymbol{Q}_{\mathrm{q}} = \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{q}_{\mathrm{q}}\}$ depending on the weighting vector $\boldsymbol{q}_{\mathrm{q}}$ (for a quadratic form based cost function)
$\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{con}}$	Biaxial contour error weighting matrix $\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{con}} = \oplus_{i=1}^{N} \mathrm{diag}\{q_{\mathrm{q}}^{\mathrm{con}}\}$ depending on the weight $q_{\mathrm{q}}^{\mathrm{con}}$ (for a quadratic form based cost function)
$\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{lag}}$	Biaxial lag distance weighting matrix $\boldsymbol{Q}_{\mathrm{q}}^{\mathrm{lag}} = \oplus_{i=1}^{N} \mathrm{diag}\{q_{\mathrm{q}}^{\mathrm{lag}}\}$ depending on the weight $q_{\mathrm{q}}^{\mathrm{lag}}$ (for a quadratic form based cost function)
R	Motor resistance
$\boldsymbol{R}_{\ell_1}$	Rate of input change weighting matrix $\boldsymbol{R}_{\ell_1} = \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\ell_1}\}$ depending on the weighting vector $\boldsymbol{r}_{\ell_1}$ (for an ℓ_1-norm based cost function)
$\boldsymbol{R}_{\mathrm{q}}$	Rate of input change weighting matrix $\boldsymbol{R}_{\mathrm{q}} = \oplus_{i=1}^{N} \mathrm{diag}\{\boldsymbol{r}_{\mathrm{q}}\}$ depending on the weighting vector $\boldsymbol{r}_{\mathrm{q}}$ (for a quadratic form based cost function)
$\boldsymbol{S}_u$	Input scaling matrix
$\boldsymbol{S}_x$	State scaling matrix
t	Time
τ_{el}	Electrical time constant
$\boldsymbol{T}^{\mathrm{con}}$	Biaxial contour error transformation matrix $\boldsymbol{T}^{\mathrm{con}}[k] = \oplus_{i=1}^{N} \boldsymbol{t}^{\mathrm{con}}[k+i]$ depending on the vector $\boldsymbol{t}^{\mathrm{con}}$
t_{final}	Time when the final state $\boldsymbol{x}_{\mathrm{final}}$ is reached (minimum-time optimal control problem)
$\boldsymbol{T}^{\mathrm{lag}}$	Biaxial lag distance transformation matrix $\boldsymbol{T}^{\mathrm{lag}}[k] = \oplus_{i=1}^{N} \boldsymbol{t}^{\mathrm{lag}}[k+i]$ depending on the vector $\boldsymbol{t}^{\mathrm{lag}}$
T_{PWM}	Sampling time of the PWM stage
T_{s}	Sampling time of the control system
$\boldsymbol{U}$	Stacked input vector (optimization variable)
$\boldsymbol{u}$	System input. For the SISO case only one axis is considered ($\boldsymbol{u} = \begin{bmatrix} u_1 \end{bmatrix}$ or $\boldsymbol{u} = \begin{bmatrix} u_2 \end{bmatrix}$); in the MIMO case two axes are combined ($\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^\top$)
$\boldsymbol{U}_{\mathrm{max}}$	Stacked maximum input vector consisting of the maximum input vector $\boldsymbol{u}_{\mathrm{max}}$
$\mathcal{U}^{\mathrm{r,opt}}$	Stacked optimized input vector consisting of the sequence of optimized input vectors $\boldsymbol{u}^{\mathrm{r,opt}}$
$\boldsymbol{U}^\star$	Optimal value of the optimization variable $\boldsymbol{U}$ (optimization result)
$\boldsymbol{u}^\star$	A single entry of the optimization result $\boldsymbol{U}^\star$

$\boldsymbol{U}^{\mathrm{tf}}(z)$	Input vector in z-domain
$\boldsymbol{V}$	State-space control gain matrix to achieve zero steady-state error
v	Motor input voltage. The specific axes are denoted by v_1 and v_2.
v_{DC}	Supply voltage
$v_{\max}$	Maximum motor input voltage
φ	Angular position. The specific axes are denoted by φ_1 and φ_2.
φ_{l}	Angular load position. The specific axes are denoted by $\varphi_{\mathrm{l},1}$ and $\varphi_{\mathrm{l},2}$.
$\varphi_{\mathrm{l}}^{\mathrm{meas}}$	Measured angular load position. The specific axes are denoted by $\varphi_{\mathrm{l},1}^{\mathrm{meas}}$ and $\varphi_{\mathrm{l},2}^{\mathrm{meas}}$.
$\varphi_{\mathrm{l}}^{\mathrm{r,opt}}$	Optimized angular load position
$\varphi_{\mathrm{l}}^{\mathrm{r}}$	Unoptimized reference for the angular load position
φ_{m}	Angular motor position. The specific axes are denoted by $\varphi_{\mathrm{m},1}$ and $\varphi_{\mathrm{m},2}$.
$\varphi_{\mathrm{m}}^{\mathrm{meas}}$	Measured angular motor position. The specific axes are denoted by $\varphi_{\mathrm{m},1}^{\mathrm{meas}}$ and $\varphi_{\mathrm{m},2}^{\mathrm{meas}}$.
$\boldsymbol{X}$	Stacked state vector (predicted)
$\boldsymbol{x}$	System state. For the SISO case only one axis is considered ($\boldsymbol{x} = \boldsymbol{x_1}$ or $\boldsymbol{x} = \boldsymbol{x_2}$); in the MIMO case two axes are combined ($\boldsymbol{x} = [\, \boldsymbol{x_1}\ \boldsymbol{x_2} \,]^\top$)
$\boldsymbol{x}_{\mathrm{final}}$	Final state $\boldsymbol{x}_{\mathrm{final}}$ (minimum-time optimal control problem)
$\boldsymbol{x}_{\mathrm{initial}}$	Initial state $\boldsymbol{x}_{\mathrm{initial}}$ (minimum-time optimal control problem)
$\boldsymbol{X}_{\max}$	Stacked maximum state vector consisting of the maximum state vector $\boldsymbol{x}_{\max}$
$\mathcal{X}^{\mathrm{r,opt}}$	Stacked optimized state vector consisting of the sequence of optimized input vectors $\boldsymbol{x}^{\mathrm{r,opt}}$
$\boldsymbol{x}_{\mathrm{pred}}$	Predicted state vector
$\boldsymbol{X}^{\mathrm{tf}}(z)$	State vector in z-domain
$\boldsymbol{Y}$	Stacked output vector (predicted)
$\boldsymbol{y}$	System output. For the SISO case only one axis is considered ($\boldsymbol{y} = [\, y_1 \,]$ or $\boldsymbol{y} = [\, y_2 \,]$); in the MIMO case two axes are combined ($\boldsymbol{y} = [\, y_1\ y_2 \,]^\top$)
$\boldsymbol{y}^{\mathrm{r}}$	Unoptimized reference vector. The specific axes are denoted by y_1^{r} and y_2^{r}.
$\boldsymbol{y}^{\mathrm{r,ff}}$	Input signal of the state-space controller and output of the feedforward part in the 2-DoF based controller design. The specific axes are denoted by $y_1^{\mathrm{r,ff}}$ and $y_2^{\mathrm{r,ff}}$.

$\mathcal{Y}^{\mathrm{r,opt}}$	Stacked optimized output vector consisting of the sequence of optimized output vectors $\boldsymbol{y}^{\mathrm{r,opt}}$
$\boldsymbol{Y}^{\mathrm{r}}$	Stacked reference vector
$\boldsymbol{Y}^{\mathrm{tf}}(z)$	Output vector in z-domain
Ψ_{pm}	Motor permanent magnet flux linkage
ω_{l}	Angular load speed. The specific axes are denoted by $\omega_{\mathrm{l},1}$ and $\omega_{\mathrm{l},2}$.
ω_{m}	Angular motor speed. The specific axes are denoted by $\omega_{\mathrm{m},1}$ and $\omega_{\mathrm{m},2}$.
ω^{path}	Path speed of a biaxial positioning system

Acronyms

2-DoF	Two-degrees-of-freedom
AC	Alternating current
ADC	Analog-to-digital converter
CCC	Cross-coupling controller
CNC	Computer numerical control
CR	Control reserve
DC	Direct current
DLL	Dynamic link library
DSP	Digital signal processor
DynFF	Dynamic, model-based feedforward control
EMF	Electromotive force
FEA	Finite element analysis
FPGA	Field programmable gate array
GPC	Generalized predictive control
IAE	Integral absolute error
ISE	Integral square error
ITSE	Integral of time multiplied by squared error
LMI	Linear matrix inequality
LP	Linear program
LQG	Linear quadratic Gaussian
LQR	Linear quadratic regulator
LTI	Linear time-invariant
LTV	Linear time-varying
MIMO	Multiple-input, multiple-output
MPC	Model predictive control
MPCC	Model predictive contouring control

MPTC	Model predictive tracking control
OS	Overshoot
PC	Personal computer
PID	Proportional-integral-derivative
PLC	Programmable logic controller
PM	Permanent magnet
PRG	Predictive reference governor
PSD	Position sensitive device
PSO	Particle swarm optimization
PWM	Pulse width modulation
QP	Quadratic program
RG	Reference governor
RLS	Recursive least squares
RMPC	Robust model predictive control
RT	Rise time
SDP	Semidefinite program
SIMO	Single-input, multiple-output
SISO	Single-input, single-output
SOCP	Second-order cone program
ST	Settling time
TF	Transfer function
ZOH	Zero-order hold
ZPETC	Zero-phase-error tracking control

List of Figures

List of Tables

List of Publications

Peer-reviewed publications of the author in chronological order:

Stumper, J.-F., Dötlinger, A., Jung, J., and Kennel, R. M. (2011). Predictive control of a permanent magnet synchronous machine based on real-time dynamic optimization. In *European Conference on Power Electronics and Applications*, pages 1–8, Birmingham, UK

Stumper, J.-F., Dötlinger, A., and Kennel, R. M. (2012). Classical model predictive control of a permanent magnet synchronous motor. *European Power Electronics and Drives Journal*, 22(3):24–31

Stumper, J.-F., Dötlinger, A., and Kennel, R. M. (2013). Loss minimization of induction machines in dynamic operation. *IEEE Transactions on Energy Conversion*, 28(3):726–735

Dötlinger, A. and Kennel, R. M. (2013b). Reference governor based model predictive contouring control for biaxial systems. In *IEEE International Conference on Industrial Technology*, pages 386–391, Cape Town, South Africa

Dötlinger, A. and Kennel, R. M. (2013a). Receding horizon based trajectory planning for biaxial systems. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 152–157, Wollongong, Australia

Dötlinger, A., Stumper, J.-F., and Kennel, R. M. (2013). Receding horizon based trajectory planning and two-degree-of-freedom tracking control for fast sampling constrained systems. In *IEEE Symposium on Predictive Control of Electrical Drives and Power Electronics*, pages 1–6, Munich, Germany

Hackl, C. M., Larcher, F., Dötlinger, A., and Kennel, R. M. (2013). Is multiple-objective model-predictive control 'optimal'? In *IEEE Symposium on Predictive Control of Electrical Drives and Power Electronics*, pages 1–8, Munich, Germany [1]

Dötlinger, A., Larcher, F., and Kennel, R. M. (2014). Robust receding horizon based trajectory planning. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 845–851, Besancon, France

[1] Hackl, C. M., Larcher, F., and Dötlinger, A. contributed equally to this work.

Dötlinger, A. and Kennel, R. M. (2014). Near time-optimal model predictive control using an L1-norm based cost functional. In *IEEE Energy Conversion Congress and Exposition*, pages 3504–3511, Pittsburgh, PA, USA

Advised Student Theses

In chronological order:

Birnkammer, F. (2013), *Minimum-Zeit Stromregelung für Gleichstrom- und Synchronmaschinen*, Bachelor's Thesis (co-supervised).

Larcher, F. (2013), *Modellprädiktive Regelung mit mehreren Optimalitätsanforderungen—Eine beispielhafte Untersuchung*, Advanced Seminar (co-supervised).

Müller, T. (2013), *Identification of Physical Plant Parameters of a Closed-Loop Control System*, Advanced Seminar.

Acikgöz, M. and Ivanov, D. (2013), *DSP basierte feldorientierte deadbeat Stromregelung einer PMSM*, Research Internship.

Birnkammer, F. (2013), *Trajectory Planning Methods for Robotics and CNC Machining*, Advanced Seminar.

Müller, T. (2013), *Parameter Identification of a Two-Mass-System in State-Space Representation*, Master's Thesis.

Larcher, F. (2013), *Robust Model Predictive Approaches for Reference Management of Fast-Sampling Systems*, Master's Thesis.

Leitner, L. (2014), *Discrete Time-Optimal Control for a Two-Mass System*, Research Internship.

Mayer, C. (2014), *Constrained Flatness based Trajectory Planning for Linear Time-Invariant Systems*, Master's Thesis.

Bauer, F. (2014), *Predictive Reference Governors for Position Control of Constrained Electrical Drives*, Master's Thesis.

Bibliography

Aghaei, S., Sheikholeslam, F., Farina, M., and Scattolini, R. (2013). An MPC-based reference governor approach for offset-free control of constrained linear systems. *International Journal of Control*, 86(9):1534–1539.

Aguilera, R. P. and Quevedo, D. E. (2011). On the stability of MPC with a finite input alphabet. In *IFAC World Congress*, pages 7975–7980, Milano, Italy.

Alders, D., Kennel, R. M., Krah, J. O., and Quan, J. (2005). Suppressing low frequency resonance oscillations of a two-mass system by active damping. *European Power Electronics and Drives Journal*, 15(4).

Aström, K., Hagander, P., and Sternby, J. (1984). Zeros of sampled systems. *Automatica*, 20(1):31–38.

Bellman, R. (1956). Dynamic programming and Lagrange multipliers. *National Academy of Sciences of the United States of America*, 10(42):767–769.

Bemporad, A. (1997). *Reference Governors: On-Line Set-Point Optimization Techniques for Constraint Fulfillment*. Dissertation, Università di Firenze.

Bemporad, A., Borrelli, F., and Morari, M. (2002a). Model predictive control based on linear programming—the explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985.

Bemporad, A. and Morari, M. (1999). Robust model predictive control: A survey. In *Lecture Notes in Control and Information Sciences*, pages 207–226. Springer Berlin Heidelberg.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. (2002b). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20.

Bemporad, A. and Mosca, E. (1995). Nonlinear predictive reference governor for constrained control systems. In *IEEE Conference on Decision and Control*, pages 1205–1210, New Orleans, LA, USA.

Bemporad, A. and Mosca, E. (1998). Fulfilling hard constraints in uncertain linear systems by reference managing. *Automatica*, 34(4):451–461.

Betts, J. T. (2001). *Practical Methods for Optimal Control Using Nonlinear Programming*. Society for Industrial and Applied Mathematics Philadelphia.

Bolognani, S., Kennel, R., Kuehl, S., and Paccagnella, G. (2011). Speed and current model predictive control of an IPM synchronous motor drive. In *IEEE International Electric Machines & Drives Conference*, pages 1597–1602, Niagara Falls, ON, USA.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Bradley, A. M. (2010). *Algorithms for the Equilibration of Matrices and their Application to Limited-Memory Quasi-Newton Methods*. Dissertation, Stanford University.

Butterworth, J. A., Pao, L. Y., and Abramovitch, D. Y. (2008). The effect of nonminimum-phase zero locations on the performance of feedforward model-inverse control techniques in discrete-time systems. In *American Control Conference*, pages 2696–2702, Seattle, WA, USA.

Cairano, S. D. and Bemporad, A. (2010). Model predictive control tuning by controller matching. *IEEE Transactions on Automatic Control*, 55(1):185–190.

Camacho, E. F. and Bordons, C. (2004). *Model Predictive Control*. Springer Berlin Heidelberg, 2. edition.

Campo, P. J. and Morari, M. (1987). Robust model predictive control. In *American Control Conference*, pages 1021–1026, Minneapolis, MN, USA.

Casavola, A., Giannelli, M., and Mosca, E. (2000a). Min-max predictive control strategies for input-saturated polytopic uncertain systems. *Automatica*, 36(1):125–133.

Casavola, A., Mosca, E., and Angeli, D. (2000b). Robust command governors for constrained linear systems. *IEEE Transactions on Automatic Control*, 45(11):2071–2077.

Chang, Y.-C. and Tsao, T.-C. (2014). Minimum-time contour tracking with model predictive control approach. In *American Control Conference*, pages 1821–1826, Portland, OR, USA.

Chen, D., Bako, L., and Lecoeuche, S. (2012). The minimum-time problem for discrete-time linear systems: A non-smooth optimization approach. In *IEEE International Conference on Control Applications*, pages 196–201, Dubrovnik, Croatia.

Cortes, P., Kazmierkowski, M. P., Kennel, R. M., Quevedo, D. E., and Rodriguez, J. (2008). Predictive control in power electronics and drives. *IEEE Transactions on Power Electronics*, 55(12):4312–4324.

Cortes, P., Kouro, S., La Rocca, B., Vargas, R., Rodriguez, J., Leon, J. I., Vazquez, S., and Franquelo, L. G. (2009). Guidelines for weighting factors design in model predictive control of power converters and drives. In *IEEE International Conference on Industrial Technology*, pages 1–7, Gippsland, VIC, USA.

Crossley, T. and Porter, B. (1974). Dead-beat control of sampled-data systems with bounded input. *International Journal of Control*, 19(5):37–41.

Dantzig, G. B. (1965). *Linear Programming and Extensions*. Princeton University Press.

Dave, P., Willig, D. A., Kudva, G. K., Pekny, J. F., and Doyle, F. J. (1997). LP methods in MPC of large-scale systems: Application to paper-machine CD control. *AIChE Journal*, 43(4):1016–1031.

Domahidi, A. (2013). *Methods and Tools for Embedded Optimization and Control*. Dissertation, ETH Zürich.

Domahidi, A., Chu, E., and Boyd, S. (2013). ECOS: An SOCP solver for embedded systems. In *European Control Conference*, pages 3071–3076, Zurich, Switzerland.

Dötlinger, A. and Kennel, R. M. (2013a). Receding horizon based trajectory planning for biaxial systems. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 152–157, Wollongong, Australia.

Dötlinger, A. and Kennel, R. M. (2013b). Reference governor based model predictive contouring control for biaxial systems. In *IEEE International Conference on Industrial Technology*, pages 386–391, Cape Town, South Africa.

Dötlinger, A. and Kennel, R. M. (2014). Near time-optimal model predictive control using an L1-norm based cost functional. In *IEEE Energy Conversion Congress and Exposition*, pages 3504–3511, Pittsburgh, PA, USA.

Dötlinger, A., Larcher, F., and Kennel, R. M. (2014). Robust receding horizon based trajectory planning. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 845–851, Besancon, France.

Dötlinger, A., Stumper, J.-F., and Kennel, R. M. (2013). Receding horizon based trajectory planning and two-degree-of-freedom tracking control for fast sampling constrained systems.

In *IEEE Symposium on Predictive Control of Electrical Drives and Power Electronics*, pages 1–6, Munich, Germany.

Ellis, G. (2000). *Control System Design Guide*. Academic Press, 2. edition.

Ellis, G. and Lorenz, R. D. (1999). Comparison of motion control loops for industrial applications. In *IEEE Industry Applications Conference*, pages 2599–2605.

Endisch, C. (2009). *Optimierungsstrategien für die Identifikation mechatronischer Systeme*. Dissertation, Technische Universität München.

Endisch, C., Brache, M., Endisch, P., Schröder, D., and Kennel, R. (2009). Identification of mechatronic systems with dynamic neural networks using prior knowledge. In *World Congress on Engineering and Computer Science*, pages 1–7, San Francisco, CA, USA.

Erkorkmaz, K. and Altintas, Y. (2001). High speed CNC system design. Part I: Jerk limited trajectory generation and quintic spline interpolation. *International Journal of Machine Tools and Manufacture*, 41(9):1323–1345.

Ferreau, H., Bock, H., and Diehl, M. (2008). An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830.

Franklin, G. F., Powell, J. D., and Emami-Naeini, A. (2002). *Feedback Control of Dynamic Systems*. Prentice Hall, 4. edition.

Franklin, G. F., Powell, J. D., and Workman, M. L. (1997). *Digital Control of Dynamic Systems*. Addison-Wesley Longman Publishing Co., 3. edition.

Fuentes, E. J., Silva, C. A., and Yuz, J. I. (2012). Predictive speed control of a two-mass system driven by a permanent magnet synchronous motor. *IEEE Transactions on Industrial Electronics*, 59(7):2840–2848.

Garriga, J. L. and Soroush, M. (2010). Model predictive control tuning methods: A review. *Industrial & Engineering Chemistry Research*, 49(8):3505–3515.

Gilbert, E. and Tan, K. (1991). Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020.

Gilbert, E. and Tan, K. T. (1995). Discrete-time reference governors and the nonlinear control of systems with state and control constraints. *International Journal of Robust and Nonlinear Control*, 5(5):487–504.

Gilbert, E. G. and Kolmanovsky, I. (1995). Discrete-time reference governors for systems with state and control constraints and disturbance inputs. In *IEEE Conference on Decision and Control*, pages 1189–1194, New Orleans, LA, USA.

Giselsson, P. (2013). Improving fast dual ascent for MPC—Part II: The embedded case. *arXiv:1312.3013*, pages 1–13.

Goodwin, G. (2012). Predictive control: A historical perspective. *International Journal of Robust and Nonlinear Control*, 22(12):1296–1313.

Graichen, K. (2006). *Feedforward control design for finite time transition problems of nonlinear systems with input and output constraints*. Dissertation, Universität Stuttgart.

Guo, H., Ohta, Y., and Masubuchi, I. (2011). Seek control of hard disk drives using reference governor under input saturation. In *Conference on Electrical and Control Engineering*, pages 4955–4958, Yichang, China.

Guzman, J., Álamo, T., and Berenguel, M. (2009). A robust constrained reference governor approach using linear matrix inequalities. *Journal of Process Control*, 19(5):773–784.

Hackl, C. M. (2012). *Contributions to high-gain adaptive control in mechatronics*. Dissertation, Technische Universität München.

Hackl, C. M., Larcher, F., Dötlinger, A., and Kennel, R. M. (2013). Is multiple-objective model-predictive control 'optimal'? In *IEEE Symposium on Predictive Control of Electrical Drives and Power Electronics*, pages 1–8, Munich, Germany.

Herceg, M., Kvasnica, M., Jones, C. N., and Morari, M. (2013). Multi-parametric toolbox 3.0. In *European Control Conference*, pages 502–510, Zurich, Switzerland.

Hirata, K. and Kogiso, K. (2001). A performance improving off-line reference management for systems with state and control constraints. In *IEEE Conference on Decision and Control*, pages 4645–4650, Orlando, FL, USA.

Hostetter, G. H. and Meditch, J. S. (1973). On the generalization of observers to systems with unmeasurable, unknown inputs. *Automatica*, 9(6):721–724.

Huber, J., Gruber, C., and Hofbaur, M. (2013). Online trajectory optimization for nonlinear systems by the concept of a model control loop—applied to the reaction wheel pendulum. In *IEEE International Conference on Control Applications*, pages 935–940, Hyderabad, India.

Isermann, R. (2005). *Mechatronic Systems: Fundamentals*. Springer Berlin Heidelberg.

Isermann, R. and Münchhof, M. (2011). *Identification of Dynamic Systems*. Springer Berlin Heidelberg.

ISO 230-4:2005 (2005). Test code for machine tools – Part 4: Circular tests for numerically controlled machine tools.

ISO 230-6:2002 (2002). Test code for machine tools – Part 6: Determination of positioning accuracy on body and face diagonals (Diagonal displacement tests).

Jerez, J. L., Goulart, P. J., Richter, S., Constantinides, G. A., Kerrigan, E. C., and Morari, M. (2013). Embedded predictive control on an FPGA using the fast gradient method. In *European Control Conference*, pages 3614–3620, Zurich, Switzerland.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45.

Karamanakos, P., Geyer, T., and Kennel, R. M. (2014a). Reformulation of the long-horizon direct model predictive control problem to reduce the computational effort. In *IEEE Energy Conversion Congress and Exposition*, pages 3512–3519, Pittsburgh, PA, USA.

Karamanakos, P., Geyer, T., Oikonomou, N., Kieferndorf, F. D., and Manias, S. (2014b). Direct model predictive control—a review of strategies that achieve long prediction intervals for power electronics. *IEEE Industrial Electronics Magazine*, 8(1):32–43.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *IEEE International Conference on Neural Networks*, pages 1942–1948, Perth, Australia.

Kirk, D. E. (2004). *Optimal Control Theory: An Introduction*. Dover Publications.

Kogiso, K. and Hirata, K. (2006). Reference governor for constrained systems with time-varying references. *Robotics and Autonomous Systems*, 57(3):289–295.

Kolmanovsky, I., Kalabic, U., and Gilbert, E. (2012). Developments in constrained control using reference governors. In *IFAC Nonlinear Model Predictive Control Conference*, pages 282–290, Leeuwenhorst, Netherlands.

Koren, Y. (1980). Cross-coupled biaxial computer control for manufacturing systems. *Journal of Dynamic Systems, Measurement, and Control*, 102(4):265–272.

Kothare, M. V., Balakrishnan, V., and Morari, M. (1996). Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379.

Kröger, T. (2010). *On-Line Trajectory Generation in Robotic Systems*. Springer Berlin Heidelberg.

Lam, D., Manzie, C., and Good, M. (2013). Model predictive contouring control for biaxial systems. *IEEE Transactions on Control Systems Technology*, 21(2):552–559.

Lambrechts, P., Boerlage, M., and Steinbuch, M. (2005). Trajectory planning and feedforward design for electromechanical motion systems. *Control Engineering Practice*, 13(2):145–157.

Landsmann, P., Stolze, P., and Kennel, R. M. (2011). Optimal switching time calculation in predictive torque control. In *IEEE International Conference on Power Electronics and ECCE Asia*, pages 923–930, Jeju, South Korea.

Linder, A., Kanchan, R., Kennel, R., and Stolze, P. (2010). *Model-Based Predictive Control of Electric Drives*. Cuvillier.

Ljung, L. (1999). *System Identification—Theory for the User*. Prentice Hall, 2. edition.

Löfberg, J. (2003). *Minimax approaches to robust model predictive control*. Dissertation, Linköping University.

Luenberger, D. (1964). Observing the state of a linear system. *IEEE Transactions on Military Electronics*, 8(2):74–80.

Lyapunov, A. M. (1992). *The General Problem of the Stability of Motion*. Taylor & Francis.

Macfarlane, S. and Croft, E. A. (2003). Jerk-bounded manipulator trajectory planning: Design for real-time applications. *IEEE Transactions on Robotics*, 19(1):42–52.

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Prentice Hall.

Mattingley, J. and Boyd, S. (2012). CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

Mittelmann, H. D. (2003). An independent benchmarking of SDP and SOCP solvers. *Mathematical Programming*, 95(2):407–430.

Morari, M. and H. Lee, J. (1999). Model predictive control: Past, present and future. *Computers & Chemical Engineering*, 23(4–5):667–682.

Nise, N. S. (2011). *Control Systems Engineering*. Wiley, 6. edition.

Olsson, H., Aström, K., Canudas de Wit, C., Gäfvert, M., and Lischinsky, P. (1998). Friction models and friction compensation. *European Journal of Control*, 4(3):176–195.

Pacas, M., Villwock, S., Szczupak, P., and Zoubek, H. (2010). Methods for commissioning and identification in drives. *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 29(1):53–71.

Parlett, B. and Reinsch, C. (1969). Balancing a matrix for calculation of eigenvalues and eigenvectors. *Numerische Mathematik*, 13(4):293–304.

Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., Mischenko, E., and Mishchenko, E. F. (1962). *The Mathematical Theory of Optimal Processes*. Pergamon Press.

Preumont, A. (2011). *Vibration Control of Active Structures*. Kluwer Academic Publisher, Dordrecht, 3. edition.

Propoi, A. I. (1963). Use of linear programming methods for synthesizing sampled-data automatic systems. *Automation and Remote Control*, 24(7):837–844.

Qin, S. and Badgwell, T. a. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764.

Radke, A. and Gao, Z. (2006). A survey of state and disturbance observers for practitioners. In *American Control Conference*, pages 5183–5188, Minneapolis, MN, USA.

Ramesh, R., Mannan, M., and Poo, A. (2005). Tracking and contour error control in CNC servo systems. *International Journal of Machine Tools and Manufacture*, 45(3):301–326.

Rao, A. V., Benson, D. A., Darby, C., Patterson, M. A., Francolin, C., Sanders, I., and Huntington, G. T. (2010). GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the gauss pseudospectral method. *ACM Transactions on Mathematical Software*, 37(2):1–39.

Rao, C. V. and Rawlings, J. B. (2000). Linear programming and model predictive control. *Journal of Process Control*, 10(2):283–289.

Rawlings, J. B. and Mayne, D. Q. (2009). *Model Predictive Control: Theory and Design*. Nob Hill Publishing.

Renton, D. and Elbestawi, M. (2000). High speed servo control of multi-axis machine tools. *International Journal of Machine Tools and Manufacture*, 40(4):539–559.

Richter, S., Jones, C. N., and Morari, M. (2012). Computational complexity certification for real-time MPC with input constraints based on the fast gradient method. *IEEE Transactions on Automatic Control*, 57(6):1391–1403.

Rodriguez, J., Kazmierkowski, M. P., Espinoza, J. R., Zanchetta, P., Haitham, A.-R., Young, H. A., and Rojas, C. A. (2013). State of the art of finite control set model predictive control in power electronics. *IEEE Transactions on Industrial Informatics*, 9(2):1003–1016.

Roppenecker, G. (2009). State feedback control of linear systems—a renewed approach. *at – Automatisierungstechnik*, 57(10):491–498.

Rothwangl, H. (2001). Numerical synthesis of the time optimal nonlinear state controller via mixed integer programming. In *American Control Conference*, pages 3201–3205, Arlington, VA, USA.

Saarakkala, S. E. and Hinkkanen, M. (2013). Identification of two-mass mechanical systems in closed-loop speed control. In *Annual Conference of the IEEE Industrial Electronics Society*, pages 2905–2910, Vienna, Austria.

Saberi, A., Lin, Z., and Teel, A. (1996). Control of linear systems with saturating actuators. *IEEE Transactions on Automatic Control*, 41(3):368–378.

Saberi, A., Stoorvogel, A., and Sannuti, P. (2000). *Control of Linear Systems with Regulation and Input Constraints*. Springer Berlin Heidelberg.

Saffer, D. R. and Doyle, F. J. (2004). Analysis of linear programming in model predictive control. *Computers & Chemical Engineering*, 28(12):2749–2763.

Schitter, G. and Astrom, K. (2007). Design and modeling of a high-speed AFM-scanner. *IEEE Transactions on Control Systems Technology*, 15(5):906–915.

Schröder, D. (2013). *Elektrische Antriebe – Grundlagen*. Springer Berlin Heidelberg, 5. edition.

Schuurmans, J. and Rossiter, J. A. (2000). Robust predictive control using tight sets of predicted states. *IEE Proceedings - Control Theory and Applications*, 147(1):13–18.

Serkies, P. J. and Szabat, K. (2013). Application of the MPC to the position control of the two-mass drive system. *IEEE Transactions on Industrial Electronics*, 60(9):3679–3688.

Stoican, F., Olaru, S., Seron, M., and De Doná, J. (2012). Reference governor design for tracking problems with fault detection guarantees. *Journal of Process control*, 22(5):829–836.

Stolze, P. and Karamanakos, P. (2013). Variable switching point predictive torque control for the three-level neutral point clamped inverter. In *European Conference on Power Electronics and Applications*, pages 1–10, Lille, France.

Stumper, J.-F. (2013). *Flatness-based predictive and optimal control for electrical drives*. Dissertation, Technische Universität München.

Stumper, J.-F., Dötlinger, A., Jung, J., and Kennel, R. M. (2011). Predictive control of a permanent magnet synchronous machine based on real-time dynamic optimization. In *European Conference on Power Electronics and Applications*, pages 1–8, Birmingham, UK.

Stumper, J.-F., Dötlinger, A., and Kennel, R. M. (2012). Classical model predictive control of a permanent magnet synchronous motor. *European Power Electronics and Drives Journal*, 22(3):24–31.

Stumper, J.-F., Dötlinger, A., and Kennel, R. M. (2013). Loss minimization of induction machines in dynamic operation. *IEEE Transactions on Energy Conversion*, 28(3):726–735.

Sturm, J. F. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1–4):625–653.

Sugie, T. and Fukui, T. (2003). Reference shaping for uncertain systems with constraints and its experimental validation. In *American Control Conference*, pages 1236–1241, Denver, CO, USA.

Sugie, T. and Yamamoto, H. (2001). Reference management for closed loop systems with state and control constraints. In *American Control Conference*, pages 1426–1431, Arlington, VA, USA.

Susanu, M. and Dumur, D. (2005). Using predictive techniques within CNC machine tools feed drives. In *IEEE Conference on Decision and Control*, pages 5150–5155, Seville, Spain.

Szabat, K., Serkies, P., Nalepa, R., and Cychowski, M. (2010). Predictive position control of elastic dual-mass drives under torque and speed constraints. In *International Power Electronics and Motion Control Conference*, pages 79–83, Ohrid, Macedonia.

Tang, L. and Landers, R. G. (2012). Predictive contour control with adaptive feed rate. *IEEE/ASME Transactions on Mechatronics*, 17(4):669–679.

Tang, L. and Landers, R. G. (2013). Multiaxis contour control—the state of the art. *IEEE Transactions on Control Systems Technology*, 21(6):1997 – 2010.

Tang, Y., Loh, H., Fuh, J., Wong, Y., and Lu, L. (2004). Accuracy analysis and improvement for direct laser sintering. In *Singapore-MIT Alliance Symposium*, pages 1–8, Singapore.

Tesfaye, A. and Tomizuka, H. (1995). Zeros of discretized continuous systems expressed in the Euler operator—An asymptotic analysis. *IEEE Transactions on Automatic Control*, 40(4):743–747.

Thomsen, S. and Fuchs, F. W. (2011). Design and analysis of a flatness-based control approach for speed control of drive systems with elastic couplings and uncertain loads. In *European Power Electronics and Drives Journal*, pages 1–10.

Tomizuka, M. (1987). Zero phase error tracking algorithm for digital control. *Journal of Dynamic Systems, Measurement, and Control*, 109(1):65–68.

Ullmann, F. (2011). *FiOrdOs: A MATLAB Toolbox for C-Code Generation for First Order Methods*. Master's thesis, ETH Zürich.

Unterholzner, A. and Wünsche, H.-J. (2009). Adaptive state space control of a camera platform for an autonomous ground vehicle. In *IEEE Intelligent Vehicles Symposium*, pages 591–596, Xi'an, China.

Van den Broeck, L., Diehl, M., and Swevers, J. (2009). Time optimal MPC for mechatronic applications. In *IEEE Conference on Decision and Control*, pages 8040–8045, Shanghai, China.

Van den Broeck, L., Diehl, M., and Swevers, J. (2010). Embedded optimization for input shaping. *IEEE Transactions on Control Systems Technology*, 18(5):1146–1154.

Van der Wal, E. (2001). Creating reusable, hardware independent motion control applications via IEC 61131-3 and PLCopen motion control profile. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 769–774, Como, Italy.

Vichik, S. and Borrelli, F. (2013). A novel method of solving linear programs with an analog circuit. *arXiv:1305.0853*, pages 1–8.

Vichik, S. and Borrelli, F. (2014). Solving linear and quadratic programs with an analog circuit. *Computers & Chemical Engineering*, 70(5):160–171.

Villwock, S. and Pacas, M. (2008). Application of the Welch-method for the identification of two-and three-mass-systems. *IEEE Transactions on Industrial Electronics*, 55(1):457–466.

Wan, Z. and Kothare, M. V. (2003). An efficient off-line formulation of robust model predictive control using linear matrix inequalities. *Automatica*, 39(5):837–846.

Waschl, H., Alberer, D., and del Re, L. (2012). Automatic tuning methods for mpc environments. In *Computer Aided Systems Theory*, pages 41–48. Springer Berlin Heidelberg.

Weck, M. (1995). *Werkzeugmaschinen, Fertigungssysteme. Band 3.2. Automatisierung und Steuerungstechnik*. Springer Berlin Heidelberg, 4. edition.

Weck, M. and Brecher, C. (2006). *Werkzeugmaschinen. Band 5. Messtechnische Untersuchung und Beurteilung, dynamische Stabilität*. Springer Berlin Heidelberg, 7. edition.

Zheng, A. (1995). *Robust Control of Systems Subject to Constraints*. Dissertation, California Institute of Technology.

Zhou, K. and Doyle, J. C. (1998). *Essentials of Robust Control*. Prentice Hall.

Zhu, K. Y. and Chen, B. P. (2001). Cross-coupling design of generalized predictive control. *Journal of Systems and Control Engineering*, 215(4):375–384.

Zirn, O. (2008). *Machine Tool Analysis — Modelling, Simulation and Control of Machine Tool Manipulators*. Habilitation thesis, ETH Zürich.

Zometa, P., Markus, K., and Findeisen, R. (2013). muAO-MPC: A free code generation tool for embedded real-time linear model predictive control. In *American Control Conference*, pages 5320–5325, Washington, DC, USA.