SLOVAK UNIVERSITY OF TECHNOLOGY
IN BRATISLAVA

FACULTY OF CHEMICAL AND FOOD TECHNOLOGY

Reg. No.: FCHPT-4622-82049

Linear Model Predictive Control Design for Complex Systems

DISSERTATION THESIS

2024 **Ing. Lenka Galčíková**

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

FACULTY OF CHEMICAL AND FOOD TECHNOLOGY

Reg. No.: FCHPT-4622-82049

Linear Model Predictive Control Design for Complex Systems

DISSERTATION THESIS

Study programme:	Process Control
Study field:	Cybernetics
Training workspace:	Institute of Information Engineering, Automation, and Mathematics
Thesis supervisor:	doc. Ing. Juraj Oravec, PhD.

2024 **Ing. Lenka Galčíková**

Slovak University of Technology in Bratislava
Department of Information Engineering and Process Control

Faculty of Chemical and Food Technology
Academic year: 2023/2024
Reg. No.: FCHPT-4622-82049

STU FCHPT

DISSERTATION THESIS TOPIC

Student: **Ing. Lenka Galčíková**
Student's ID: 82049
Study programme: Process Control
Study field: Cybernetics
Thesis supervisor: doc. Ing. Juraj Oravec, PhD.
Head of department: doc. Ing. Martin Klaučo, PhD.
Workplace: Institute of Information Engineering, Automation, and Mathematics, FCHPT STU

Topic: **Linear Model Predictive Control Design for Complex Systems**

Language of thesis: English

Specification of Assignment:

Linear Model Predictive Control Design for Complex Systems

The main goal of this thesis is to develop the methods of linear model predictive control (MPC) design for complex systems, including large-scale systems and/or systems with challenging properties.
The particular objectives are:
(i) to develop the MPC design approaches improving the control performance or application range of the control strategy;
(ii) to validate and analyze the developed MPC design method using numerical simulation or laboratory implementation;
(iii) develop freely available software for design of a selected control method.

Length of thesis: 100

Deadline for submission of Dissertation thesis: 31. 05. 2024
Approval of assignment of Dissertation thesis: 29. 05. 2024
Assignment of Dissertation thesis approved by: prof. Ing. Miroslav Fikar, DrSc. – Chairperson of Field of Study Board

Acknowledgment

22 years of studies. 9 years at the university. 7 years of work under the supervision of Juraj Oravec. The following lines do not even come close to describing how grateful I am for a supervisor like you. I would like to thank you for all your help, patience, many, many, many hours in the lab, and many more hours of scientific discussions and developing new ideas. Thank you for giving me enough freedom to choose the topics I wanted to explore. Thank you for all the experiences that you brought me to, including organizing the Student scientific conference. Thank you also for all the informal conversations, the PhD meetings, Exploding Kittens, and much more. Most importantly, thank you for being a teammate, not a boss. I wish you a lot of strength in staying such a mega-super-visor and mega-super-scientist for the rest of your career.

Miška, my PhD group teammate and officemate, thank you for all your help, scientific cooperation, guidance, conversations, spinning classes, dog therapy, and all our adventures.

I would like to mention and thank some of my fellow fighters on this 9-year-long journey – Kika and Roman. Thank you for making this crazy ride a bit more bearable, thanks to all the laughter (through tears) and mutual help.

I would like to thank all the remaining colleagues from our department whom I did not name. Thank you for teaching me so much.

Marek, thank you for your help, support, and love in my breakthrough life chapter. Thank you for being my inspiration.

Majuš, Sťopa, Apple, ďakujem vám za všetko. Ďakujem, že ste.

Abstract

This dissertation thesis is devoted to the linear model predictive control (MPC) design for complex systems. The term "complex system" includes two groups of challenging systems – large-scale systems and systems with nonlinear and asymmetric dynamics. The first part of the thesis deals with the partial explicit MPC design for large-scale systems, as solving large-scale optimization problems is computationally challenging. One way to deal with computational complexity is to construct offline a partial explicit MPC solution and use it in the online phase to streamline evaluating the optimal control action. The thesis proposes to replace the polytopic critical regions having variable number of halfspaces with the maximal volume inner approximations based on the Chebyshev balls. As such approximation has a fixed and known structure, the memory footprint of the partial parametric solution is also fixed and known in advance, which enables scaling the solution a priori to meet the control unit limitations. As the large-scale polytopes are not stored, the memory footprint of the partial explicit solution is significantly reduced. The second part of the thesis aims to develop a self-tunable explicit MPC for nonlinear and asymmetric systems. Including a nonlinear model in the MPC optimization problem would lead to high computational complexity and non-convex critical regions in the parametric solution. Therefore, this thesis considers a linear prediction model, and the plant nonlinearity is compensated by designing an appropriate online tuning technique. Moreover, the proposed self-tuning technique also addresses asymmetric plant dynamics (different behavior when the process variable rises and decreases). The self-tuning method is performed automatically online and depends on the reference value of the controlled variable, as this approach is focused on reference-tracking applications. The tuning method was applied to a laboratory heat exchanger, and the control performance improved compared to a non-tunable controller. The third part of the thesis focuses on the implicit tube MPC, which is suitable for large-scale uncertain systems. This thesis's contribution was the incorporation of the recent method of implicit tube MPC into a freely available MATLAB toolbox, MPT+, to spread the possibility of implementing this robust MPC technique. This thesis presents the design procedure of the implicit tube MPC in MPT+ and validates the toolbox on a large-scale system where the original tube MPC was impossible to implement due to computationally demanding geometric operations.

Abstrakt

Táto dizertačná práca je venovaná návrhu lineárneho modelu prediktívneho riadenia (MPC) pre zložité systémy. Pojem "zložitý systém" zahŕňa dve skupiny komplikovaných systémov – veľkorozmerné systémy a systémy s nelineárnou a asymetrickou dynamikou. Prvá časť práce sa zaoberá návrhom parciálneho explicitného MPC pre veľkorozmerné systémy, pretože riešenie veľkorozmerných optimalizačných problémov je výpočtovo náročné. Jedným zo spôsobov, ako sa vysporiadať s výpočtovou náročnosťou, je zostrojiť parciálne explicitné riešenie MPC a použiť ho v online fáze na zefektívnenie vyhodnocovania optimálneho akčného zásahu. Práca navrhuje nahradiť polytopické kritické regióny s premenlivým počtom polpriestorov ich vnútornou aproximáciou založenou na kruhoch s maximálnym objemom. Keďže aproximácia má fixovanú a známu štruktúru, veľkosť pamäte parametrického riešenia je tiež fixovaná a známa vopred, čo umožňuje naškálovať vopred veľkosť riešenia tak, aby vyhovovala limitáciám riadiacej jednotky. Keďže veľkorozmerné polytopy nie sú ukladané, veľkosť pamäte parciálneho explicitného riešenia je významne znížená. Druhá časť práce sa zameriava na vývoj samoladiteľného explicitného MPC pre nelineárne a asymetrické systémy. Zahrnutie nelineárneho modelu do optimalizačného problému MPC by viedlo k vysokej výpočtovej zložitosti a nekonvexným kritickým regiónom parametrického riešenia. Preto táto práca uvažuje lineárny predikčný model a nelinearita systému je kompenzovaná návrhom vhodnej techniky online ladenia. Okrem toho, navrhovaná technika samoladenia adresuje aj asymetrickú dynamiku systému (rôzne správanie, keď procesná veličina stúpa a klesá). Metóda samoladenia sa vykonáva automaticky online a je závislá od referenčnej hodnoty riadenej veličiny, keďže tento prístup je zameraný na aplikácie úlohy sledovania referencie. Metóda ladenia bola aplikovaná na laboratórny výmenník tepla a kvalita riadenia sa zlepšila v porovnaní s neladeným regulátorom. Posledná časť práce je zameraná na MPC založené na implicitných tubách, ktoré je vhodné pre veľkorozmerné neurčité systémy. Prínosom tejto práce bolo začlenenie implicitného tube MPC do voľne dostupného MATLAB toolboxu, MPT+, na rozšírenie možnosti implementácie tejto robustnej techniky MPC. Táto práca prezentuje postup návrhu implicitného tube MPC v MPT+ toolboxe a validuje toolbox na veľkorozmernom systéme, kde pôvodný prístup tube MPC nebolo možné implementovať kvôli výpočtovo náročným geometrickým operáciám.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, the application of optimal control techniques is widely expanding [59]. The implementation of the optimal control methods, e.g., model predictive control (MPC), has many significant benefits. It allows us to predict future system behavior and obtain optimal control actions based on the requirements on control, e.g., minimizing the control costs, maximizing the profits, reducing the impact on the environment, or improving the control performance. Moreover, the possibility to include constraints on the input, output, and state variable is nonnegligible. As MPC is a receding horizon control strategy, the optimization of control actions is performed in each control step in order to reduce the effect of external and internal disturbances [59]. Due to its many practical benefits, model predictive control became a widely used control strategy in the past three decades, see, e.g. [59], [14], [60], and references therein. Some form of MPC-based control is present in approximately 90 % of industrial implementation of multivariable control, see [48], [65].

However, the necessity to solve an optimization problem in each control step is a very challenging task due to the computational complexity. The challenges of the real-time MPC applications include complex systems with many constraints, e.g., robust and stochastic MPC design [53], distributed control [44], problems with a long prediction horizon, and a high number of states and control actions [15].

One of the ways to implement MPC despite its demands on real-time computational complexity is an explicit solution using multiparametric programming, see, e.g., [5], [10]. The essence of the explicit model predictive control lies in the division of implementation into two separate phases. First, in the offline phase, the controller is constructed. Particularly, the optimization problem is computed for a predefined set of parameter values, and the corresponding control law is determined. For a multiparametric quadratic problem (mpQP), the control law has the form of a piecewise affine (PWA) function over a polytopic partition composed of a set of convex critical regions. Next, in the online phase, i.e., the optimal control action is evaluated from control law

corresponding to one of the critical regions. The critical region associated with the current measurement is identified by solving the point location problem in some form of a lookup table [24].

Although the application range of explicit MPC is wide, two interconnected issues arise from its implementation: memory consumption and runtime effort. One of the possibilities of memory burden reduction is a regionless explicit model predictive control presented in [42]. The authors showed that the geometric construction and storage of the critical regions are not required. Instead, the active sets are considered, which provides significant savings in memory preserving the optimal solution. Other efficient constructions of the explicit partition using the dynamic programming were introduced in [61], [58]. The memory savings were achieved in [40], using the clipping function eliminating the number of regions of the PWA function over which the control law attains a saturated value. The utilization of the large set of critical regions evaluating the saturated control law is removed using the polynomial separator function in [41]. The polynomial separator was replaced by various convex sets in [62].

The complexity reduction techniques do not target only memory footprint but also accelerating the evaluation of the optimal control action in the online phase. One of the methods speeding up the online phase was suggested in [27], where the critical regions are sorted based on the minimal or maximal value of the corresponding value function. Using the proposed smart order, real-time control is significantly accelerated on average. Another technique leading to the decreased computational effort was an online removal of inactive constraints introduced in [30]. In [39], [79], [24] the online runtime is reduced by simplifying the point location problem without sacrificing closed-loop performance. The irrelevant critical regions are removed using the reachability analysis.

Several works bridge the gap between optimization-less real-time implementation of explicit MPC and implicit (non-explicit) MPC suitable for large-scale systems. Many later works were inspired by [17], where the online solution of the MPC problem was accelerated using the warm-start strategy based on the knowledge of the optimal active set from the previous control step. An efficient approach of semi-explicit MPC was presented in [20]. This method is based on the offline state-dependent parametrization of the optimization variables using the tailored subspace clustering algorithm and the training data consisting of the MPC optimization problem solutions. A semi-explicit approach was introduced into the move-blocking-based MPC design in [77], where the time-varying blocking structure also guarantees the recursive feasibility and closed-loop system stability. Learning approximate semi-explicit MPC for a hybrid system was designed in [51]. In [83], the real-time suboptimal MPC was designed combining the

approximated explicit solution of the MPC problem used for warm-start of the active set method.

The trade-off between the benefits and limitations of the explicit MPC implementation for the large-scale MPC problems is well-balanced in [32]. This work presents a novel perspective concept of a partial multiparametric solution which places it on the road between explicit MPC and implicit MPC. Without loss of optimality, closed-loop system stability, and recursive feasibility of the large-scale MPC problem, the partial multiparametric solution utilized in the framework of the explicit MPC improves initialization of the hot-start strategy for the real-time implementation. First, in the offline phase, a partial solution of an explicit MPC optimization problem is evaluated. The positions of the critical regions are represented by the centers of the Chebyshev balls. In the online phase, when the measurement is obtained, the critical region with the nearest center of the Chebyshev ball is identified. This critical region is used in the hot-start strategy to initialize searching for the corresponding optimal control action.

One of the contributions of this thesis is to push the idea towards the fixed-memory parametric solution of the partial explicit MPC. The term "fixed-memory" denotes that the size of the memory footprint necessary to store the parametric solution is determined in advance, i.e., before solving the multiparametric optimization problem. Inspired by the method presented in [32], the crucial idea of this contribution is not to store the polytopic critical region, but only its maximal volume inner approximation using the Chebyshev ball, i.e., data defining its center and radius. Just these data will be used in the online phase for the hot-start strategy. As the critical polytopic regions do not have the same number of halfspaces, the data size needed to be stored is not known in advance. On the other hand, storing the Chebyshev ball approximation provides us with fixing the memory of each considered critical region. As a consequence, the partial solution of explicit MPC is fixed in advance. It enables scaling the size of the solution a priori without the necessity to solve an optimization problem. Moreover, compared to [32], the proposed method significantly reduces the memory consumption of the partial solution, as the storage of the Chebyshev balls, i.e., centers and radii, requires much lower memory compared to the storage of the polytopes. Finally, the proposed method also improves the initialization of the hot-start procedure for solving the large-scale optimization problem. In contrast to [32], not only the centers of Chebyshev balls are considered, but also their radii, which provide more accurate information to identify the nearest criticalregion for the hot-start strategy.

Another challenge associated with the explicit MPC is the ability to adjust the controller setup online. The explicit MPC is not tunable in default as the conventional approach introduced in [5] considers the penalty matrices with fixed structure and values. The

inability to tune the explicit controller online can be a disadvantage due to varying operating conditions when the different setups of the controllers are beneficial.

The possibility to tune the explicit MPC online came with the publishing of [4]. The tuning parameter penalizing the control inputs became a parameter, for which the optimal controller was precomputed. Nevertheless, the application was limited only to linear cost functions of the optimization problem. To satisfy the demands for often-used quadratic cost functions, the approximated tunable explicit MPC was presented in [36]. The technique is based on two explicit model predictive controllers which differ in the setup of one penalty matrix. The two explicit MPCs provide upper and lower boundary optimal controllers. Based on the evaluation of the two boundary control inputs, the tuned control input is calculated by linear interpolation. The follow-up work [63] provided stability and recursive feasibility guarantees by proper choice of the terminal penalty matrix and terminal set constraint [55]. Moreover, the strategy in [63] extends the tuning ability based on any penalty matrix and not just the input penalty.

The idea of approximated tunable MPC with neural networks is presented in [34]. To ensure the tuning property, the penalty matrices were included in the training process. As a result, it was possible to tune the neural network-based controller online, while mimicking the nearly optimal MPC. In [33], the neural network-based tunable controller MPC was extended with a corrector which steered the controller such that the constraints on the manipulated and process variables were satisfied.

Our paper [19] pushes the idea of tunable explicit MPC further and deals with the issues of practical industrial-oriented implementation. In numerous practical applications, the reference value of the controlled variable is changed and acquires values from a wide range of operating conditions. The use of different controller setups can help handle the plant's nonlinear behavior, without the necessity to explicitly consider a nonlinear prediction model. Considering more accurate, but nonlinear prediction model would lead to higher computational complexity resulting from nonlinear parametric programming [64]. The paper [19] presents a procedure of the self-tunable controller technique. The controller's aggressivity is tuned based on the difference between the reference value and the steady state corresponding to the model linearization point. In the context of MPC, the aggressiveness is associated with the setup of the penalty matrices, as it determines the aggressiveness of the final control input. In general, higher penalization of the controlled states or control error in the cost function leads to more aggressive control actions. This process is analogous to increasing the proportional gain in the PID (proportional–integral–derivative) controller. On the contrary, higher penalization of the input variable leads to more sluggish control, e.g.,

see [50]. In [19], the MPC tuning based on the distance from the steady-state operating point represented a way how to compensate for the system's nonlinear behavior.

This thesis directly extends our early results presented in [19], where the basic principles of the self-tunable approximated explicit MPC were introduced. In this thesis, a novel method of self-tuning parameter setup is introduced. Compared to [19], the self-tuning method is based on the size of the reference step change. Moreover, the idea of further scaling of the tuning parameter is elaborated. The interval of the values of the self-tuning parameter is split at some certain value and each part of the interval corresponds to the specific operating conditions defined by the control engineer. In such a way, the complex system dynamics are addressed, e.g., the system's asymmetric behavior is compensated (different behavior when the process variable is rising and decreasing). Finally, to investigate the benefits of the proposed approach, the proposed self-tuning control policy was implemented to control a laboratory-scaled counter-current plate heat exchanger. This thesis provides the control performance evaluation and analysis using the self-tunable controller compared to the conventional boundary explicit MPCs.

Although both implicit and explicit MPC have been intensively studied in the past decades, there are still challenges worth addressing to spread its implementation. Such a development is highly dependent on tailored validation tools. The MPC design problem is addressed in plenty of well-developed software tools. This thesis focuses specifically on the MATLAB programming environment, but in [37, 49, 29], and the references therein, one can find the review on the solvers and tools supporting MPC design and its evaluation also using other programming environments, e.g., Python and Julia.

The MATLAB commercial MPC Toolbox [52] addresses various classes of the MPC design problems, including the construction of the adaptive, explicit, gain-scheduled, and nonlinear MPC controllers. This toolbox provides several built-in solvers and also a dedicated user interface MPC Designer App for the controller tuning. The nonlinear MPC design problems are efficiently solved using the ACADO Toolkit [28]. The non-convex optimization problem is solved by the sequential quadratic programming (SQP) approach. ACADO Toolkit evaluates the library-dependent C/C++ code and provides an interface for MATLAB. CasADi [3] represents another toolbox suitable for the nonlinear MPC design. This open-source package also provides interfaces for MATLAB and Python. The nonlinear MPC can be designed also using the open-source toolbox MATMPC [11]. The YALMIP [47] toolbox is a widely-used modeling parser focused on the various classes of optimization problems, including non-convex optimization. YALMIP provides also support for the MPC design problems. The Multi-Parametric Toolbox (MPT) [22] for MATLAB represents a widely-used software package for the

implicit (non-explicit) and explicit MPC design, multi-parametric optimization, and operations over convex sets. MPT integrates many tools enabling efficient construction, tuning, and validation of the advanced MPC controllers.

Besides the above mentioned topics of memory size reduction techniques (i.e., the partial explicit MPC), and strategies improving the control performance, (i.e., the self-tunable explicit MPC), there is another relevant topic worth interest – the uncertainties affecting the controlled system. Including the disturbance in the control design leads to robust control strategies. In terms of robust MPC methods, one of the most essential works is the min-max approach [75, 66]. This robust control technique includes the bounded disturbance in the prediction model, which leads to improved performance with stabilizing properties. On the other hand, the min-max MPC may lead to significant computational demands due to exponential growth of the possible predicted future scenarios.

A very widely used and accepted robust MPC approach is the tube MPC [56, 72]. Instead of considering all the future combinations of possible disturbances, this approach introduces the optimization of the initial nominal state, and robustifies the MPC optimization problem via introducing the robust positively invariant set, i.e., the tube. Many works follow up on this topic and develop improvements, e.g., in the context of homothetic tubes [69] and elastic tubes [70]. Another extensions focus on state estimation with output feedback tube MPC [54], reference tracking problem [45], parallel explicit tube MPC [82], etc.

Despite its benefits, the tube MPC comes with a challenge of applicability in higher dimensions of control probems due to non-trivial set operations. A relatively novel contribution focuses on an implicit tube MPC [67], suitable for large-scale optimization problems. This approach avoids the geometric set-based operations which extends the applicability range of the tube MPC method.

The interest in tube MPC applications and research supports the idea of providing a user-friendly tube MPC design tool. Such a tool should streamline the procedure of tube MPC design and implementation. Moreover, from the research point of view, being able to easily construct the tube MPC allows one to focus on further extensions of the original control method. In [26] and [25], the authors follow up on the functionalities of the original MATLAB MPT toolbox and, among other extensions, bring the possibility to design and implement the tube MPC in a user-friendly way, using a novel MPT+ toolbox [2]. This thesis steps further towards the implicit tube MPC design. The aim is to include the possibility to construct the tube MPC also in an implicit manner to enable a wide and user-friendly application of this control technique.

To conlude, this dissertation thesis deals with three techniques of linear MPC for complex systems. Throughout this thesis, the term “complex system” denotes the systems that bring two groups of challenges in MPC design: large-scale systems and systems with nonlinear and asymmetric dynamics. The considered control techniques are (i) partial explicit MPC, (ii) tunable approximated explicit MPC, and (iii) implicit tube MPC.

General Objectives

This dissertation thesis focuses on the linear MPC design for complex systems. The aims of this thesis are summarized in three main areas: (i) theoretical contributions in linear MPC design methods, (ii) practical validation of the proposed theoretical contributions, and (iii) freely available software development for the control of large-scale systems. More specifically, the objectives of the dissertation thesis are:

- **Theoretical contributions:**
  - **Memory footprint reduction associated with large-scale MPC.** The method of the partial explicit MPC is revisited, and the size of the memory footprint to store the partial explicit MPC is reduced without inducing suboptimality.
  - **Fixed size of memory footprint associated with large-scale MPC.** The memory footprint necessary to store the partial explicit MPC data is unpredictable. In large-scale systems control applications, it becomes a crucial question. In this thesis, fixing the memory footprint size is addressed.
  - **Elaboration of the self-tuning technique for tunable explicit MPC to improve control performance.** This thesis also focuses on the topic of control performance improvement and targets the tunable approximated explicit MPC. The techniques to tune the explicit model predictive controller automatically during the real-time control are elaborated. The proposed self-tuning strategies deal with nonlinear and asymmetric plant behavior.
- **Practical validation:**
  - **Numerical case studies on large-scale systems.** The proposed partial explicit MPC improvement is analyzed on the set of large-scale systems.
  - **Laboratory validation on a heat exchanger plant.** The proposed self-tuning strategy is implemented on a laboratory heat exchanger. The control performance is analyzed and compared to non-tuned controllers.
- **Freely available software:**
  - **Software development for control of large-scale systems.** The recent perspective approach of implicit tube MPC, suitable for controlling large-scale systems, is incorporated into the MATLAB MPT+ toolbox to spread the wide usage of this robust control technique.

– **Case study validating the developed software.** The developed MPT+ extension enabling the implicit tube MPC application is validated on a large-scale reactor-separator system.

Chapter 2

Fixed-memory partial explicit MPC

This chapter is devoted to the topic of fixed-memory partial explicit MPC. This control approach addresses the first group of complex systems considered in this thesis – the large-scale systems. First, this chapter provides an overview of the topics of model predictive control as well as its explicit solution. The theoretical overview also introduces the partial solution of explicit model predictive control, which is useful for handling large-scale optimization problems. Next, the memory footprint of the data that needs to be stored for the online phase is analyzed. Furthermore, a novel concept leading to memory footprint reduction is proposed. The polytopic critical regions are no longer stored. Instead, their maximal volume inner approximations are considered. As the inner approximation has a fixed structure, the total memory footprint of the solution is also fixed. Finally, the memory footprint of the partial solution based on the novel ideas is analyzed.

2.1 Model predictive control

Model predictive control (MPC) is a control method utilizing optimization in order to deliver the best decision, i.e., optimal control action, at the current time. The main concept is to utilize a dynamic model of the controlled system to predict its future behavior [73]. This control approach allows us to achieve significant safety improvement of production operation, control performance, minimize costs, and negative impact on the environment. These are some of the main reasons why MPC gained its popularity in the past 3 decades [57].

The solution of an optimization problem is periodically re-solved in every time sample T_s. In every control step, the whole trajectory of optimal control actions is calculated for N steps forward, but only the first value of the input variable is implemented. In the next control step, the optimization problem is solved again, see Figure 2.1. As a

consequence of this so-called receding horizon nature, the effect of external disturbances and plant-model mismatch is reduced [57].

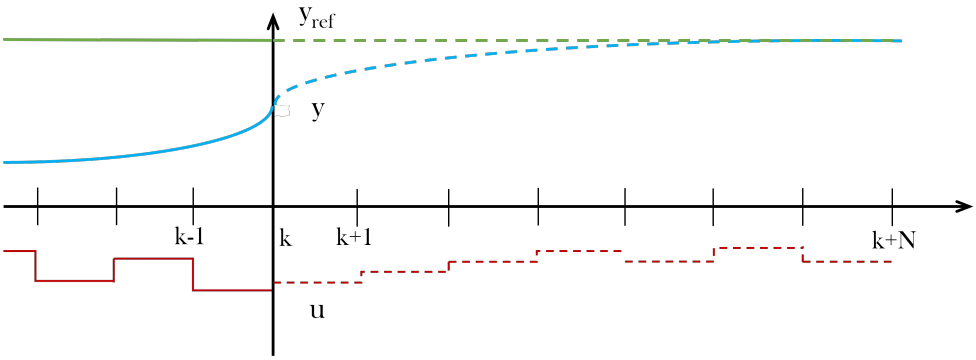


Figure 2.1: Principle of MPC. The dashed lines represent the prediction and the solid lines represent the past control steps. The red color corresponds to the trajectory of control inputs u. The controlled variable y is depicted in blue. The green line represents the reference value y_{ref}.

Another important characteristic of MPC is the ability to handle multiple-inputs and multiple-outputs (MIMO) systems, compared to well-known proportional-integral-derivative (PID) controllers. As MPC is model-based control strategy, the dynamic model is therefore essential. State-space as well as input-output prediction models are nowadays included in the control design. However, the state-space model is nowadays the most utilized choice in model predictive control technique [35].

Moreover, there is a possibility to include constraints on the input, output, and state variables to ensure safety or meet requirements on control performance. This is a significant benefit compared to linear-quadratic (LQ) optimal controllers. As a consequence, some form of MPC-based control is present in approximately 90 % of industrial implementation of multivariable control, see e.g. [48], [65].

2.1.1 Regulatory problem

Model predictive controller solves an optimization problem in every control step. In order to satisfy specific requirements on control performance and to achieve the set goals, various forms of the optimization problem can be utilized. When the aim is to push the system states to the origin from a nonzero initial condition, the regulatory

problem is solved.

In this thesis, the model predictive control is based on the following formulation of the quadratic programming (QP) optimization problem [59]:

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k), \tag{2.1a}$$

$$\text{s.t.} \quad x_{k+1} = A_\mathrm{d} x_k + B_\mathrm{d} u_k, \tag{2.1b}$$

$$u_k \in \mathbb{U}, \tag{2.1c}$$

$$x_k \in \mathbb{X}, \tag{2.1d}$$

$$x_0 = x(t), \tag{2.1e}$$

where N is the length of prediction horizon, t is time, k indicates the step of prediction horizon and acquires the values $k = 0, \dots N-1$, $x \in \mathbb{R}^{n_x}$ is the system state vector, and $u \in \mathbb{R}^{n_u}$ is the vector of the input variable. $A_\mathrm{d} \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$ represents the discrete-time system state matrix, and $B_\mathrm{d} \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{u}}$ is the discrete-time input matrix. The sets $\mathbb{U} \subseteq \mathbb{R}^{n_\mathrm{u}}$, $\mathbb{X} \subseteq \mathbb{R}^{n_\mathrm{x}}$ are convex polytopic sets of physical constraints on inputs and states, respectively. These sets include the origin in their strict interiors. The positive semi-definite matrix $Q \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}, Q_\mathrm{y} \succeq 0$ penalizes the system states, and the positive definite matrix $R \in \mathbb{R}^{n_\mathrm{u} \times n_\mathrm{u}}, R \succ 0$ penalizes the control actions [59].

By minimizing the squared value of the system states in (2.1a), the system states are pushed to the origin. Also, the squared value of control inputs makes the controller decrease the control costs. By increasing the weight on system states, the controller becomes more aggressive as the objective is to minimize the cost function where the terms consisting of weighted system states play the significant role. On the contrary, if the weight on control inputs is higher compared to the weight on system states, the control trajectory is generally sluggish. In this case, the terms consisting of the squared control inputs represent the more significant contribution to the value of the cost function. Therefore, the matrices Q and R represent tuning parameters to affect control performance [59].

From the control point of view, it is important not only to tune the weight matrices, but also to have available an accurate model in (2.1b). As a consequence, the prediction of the future states is more accurate and the effect of plant-model mismatch is reduced. Note, also output variable $y \in \mathbb{R}^{n_y}$ can be predicted for the whole prediction horizon

from well-known relation

$$y_k = C_\mathrm{d} x_k + D_\mathrm{d} u_k, \tag{2.2}$$

where $C_\mathrm{d} \in \mathbb{R}^{n_\mathrm{y} \times n_\mathrm{x}}$ and $D_\mathrm{d} \in \mathbb{R}^{n_\mathrm{y} \times n_\mathrm{u}}$ are the discrete-time state-space matrices. The state-space model is the most commonly used, as it handles relations between all the state variables [35]. Nevertheless, also input-output model representations are used for the prediction of the future system outputs.

Constraints on system states and control actions stated in (2.1c) and (2.1d) are determined by technical parameters of controlled plant or by requirements on safety or control performance. In some applications, it is sufficient to include the constraints only on the output variable instead of the system states, e.g., when the input-output model is utilized. Then, the corresponding constraint is formulated analogously to (2.1c) and (2.1d):

$$y_k \in \mathbb{Y}, \tag{2.3}$$

where the set $\mathbb{Y} \subseteq \mathbb{R}^{n_\mathrm{y}}$ is also a convex polytope and contains the origin in its strict interior.

The last constraint (2.1e) of the optimization problem assigns the value of the current measurement or estimation to a system state initial condition. This value is substituted into (2.1e) in every control step, what makes MPC the receding control strategy.

In some practical applications, a stability guarantee is required. One of the ways how to achieve stability is penalization and constraining the system states in the N-th step of the prediction horizon. Including the last step of the prediction horizon into the cost function is denoted as terminal cost or terminal penalty. When the system states at the end of the prediction horizon are forced to lie in some desired region, we denote it as a terminal set.

The optimization problem with stability guarantee acquires the following form [55]:

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) + x_N^\top P x_N, \tag{2.4a}$$

$$\text{s.t.} \quad x_{k+1} = A_\mathrm{d} x_k + B_\mathrm{d} u_k, \tag{2.4b}$$

$$u_k \in \mathbb{U}, \tag{2.4c}$$

$$x_k \in \mathbb{X}, \tag{2.4d}$$

$$x_0 = x(t), \tag{2.4e}$$

$$x_N \in \mathbb{X}_\mathrm{N}, \tag{2.4f}$$

where the matrix $P \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}, P \succ 0$ penalizes the squared value of system states at the end of the prediction horizon. The matrix P is chosen such that the terminal penalty term has a form of a Lyapunov function, and is given by the solution of the discrete-time algebraic Riccati equation [43]:

$$A_\mathrm{d}^\top P A_\mathrm{d} - P - A_\mathrm{d}^\top P B_\mathrm{d} (B_\mathrm{d}^\top P B_\mathrm{d} + R)^{-1} B_\mathrm{d}^\top P A_\mathrm{d} + Q = 0. \tag{2.5}$$

The set $\mathbb{X}_\mathrm{N} \subseteq \mathbb{R}^{n_\mathrm{x}}$ denotes the terminal set in which the system states are desired to belong in the last step of the prediction horizon. When the regulatory problem is solved, the terminal set must contain the origin. When the stability guarantee is required, the terminal set $\mathbb{X}_\mathrm{N}$ needs to be constructed as the maximal positively invariant set [6], which is obtained from the system $x_{k+1} = (A_\mathrm{d} + B_\mathrm{d}) x_k$, under the terminal feedback control law $u_k = K x_k$. The gain K is the stabilizing controller gain such that $(A_\mathrm{d} - B_\mathrm{d} K)$ is stable, e.g., LQR controller gain, which is given as follows:

$$K = -(B_\mathrm{d}^\top P B_\mathrm{d} + R)^{-1} B_\mathrm{d}^\top P A_\mathrm{d}. \tag{2.6}$$

Note, the prediction horizon should be sufficiently large to result in a feasible sequence of control inputs leading the states to the terminal set $\mathbb{X}_\mathrm{N}$ in N steps. If the terminal penalty has a form of Lyapunov function and the terminal set is chosen as a maximal positively invariant set, e.g., LQR-based set, the closed-loop stability is guaranteed [55].

2.1.2 Tracking problem

Besides the regulatory problem described in the Section 2.1.1, it is also possible to formulate the MPC optimization problem as a tracking problem. The objective is to

achieve a nonzero reference value of the system states or outputs. One of the tracking problem forms is the incremental formulation, exploiting the relation between the current and the previous control action:

$$\Delta u_k = u_k - u_{k-1}. \tag{2.7}$$

The increment of the control input is penalized and represents the optimized variable. The optimization problem acquires the following form

$$\begin{aligned}
\min_{\Delta u_0,\dots,\Delta u_{N-1}} \quad & \sum_{k=0}^{N-1} ((x_{\mathrm{ref}} - x_k)^\top Q (x_{\mathrm{ref}} - x_k) + \Delta u_k^\top R \Delta u_k), && \text{(2.8a)}\\
\text{s.t.} \quad & \widetilde{x}_{k+1} = \widetilde{A}_{\mathrm{d}} \widetilde{x}_k + \widetilde{B}_{\mathrm{d}} \Delta u_k, && \text{(2.8b)}\\
& x_k \in \mathbb{X}, && \text{(2.8c)}\\
& u_k \in \mathbb{U}, && \text{(2.8d)}\\
& \Delta u_k \in \mathbb{U}_\Delta, && \text{(2.8e)}\\
& \widetilde{x}_0 = \widetilde{x}(t). && \text{(2.8f)}
\end{aligned}$$

In the optimization problem (2.8), the state vector $\tilde{x}_k^\top = \begin{bmatrix} x_k^\top & u_{k-1}^\top \end{bmatrix}$ is augmented because of the combination of the original model constraint (2.1b) and the equation for computing the new control action based on the previous input variable and its change in (2.7). Then the augmented model in (2.8b) has the following form [57]:

$$\widetilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \begin{bmatrix} A_{\mathrm{d}} & B_{\mathrm{d}} \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} B_{\mathrm{d}} \\ I \end{bmatrix} \Delta u_k. \tag{2.9}$$

The augmented state vector changed in the first constraint (2.8b) which defines the system model, and the last one (2.8f), which defines the initial condition. The model constraint (2.8b) is just abbreviated notation of (2.9). The constraint on the change of the input variable is included in (2.8e), where the set $\mathbb{U}_\Delta \subseteq \mathbb{R}^{n_{\mathrm{u}}}$ is a convex set containing the origin in its strict interior. The remaining two constraints in (2.8d) and (2.8c) are the same as in the original formulation of regulation problem in (2.1), thus the values of the state and the input variables can be limited [57].

2.2 Explicit model predictive control

Model predictive control is often applied in practice thanks to its many benefits. As it solves a complex optimization problem in each control step, this type of control is time and memory demanding. Therefore, it involves sufficient sampling time and devices with control units that dispose of sufficient computational performance. The need for wide optimal control implementation for the systems with fast dynamics or less performing control units lead to the formation of explicit model predictive control [5].

In the fundamental work by Bemporad et al. [5], the authors show how to move all the computations necessary for the implementation of MPC offline, while preserving all its other characteristics. The entity of explicit model predictive control lies in the division of the computation and implementation of control into two phases: offline phase and online phase, see Figure 2.2. In the offline phase, the explicit model predictive controller is constructed. Particularly, the set of optimal control actions is calculated for the whole feasible set of values – polytopic partition, i.e., for all the states for which the optimization problem has a solution. Such control law, i.e., the relation between the optimal control action and the current system state, is stored in the form of some lookup table. In the online phase, the real time control is provided. By searching in the lookup table, the corresponding control law is assigned to the current system state and evaluated in every control step [73].

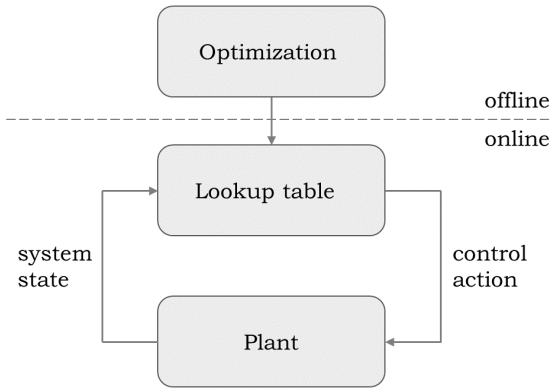


Figure 2.2: Scheme of explicit MPC.

Construction of the explicit model predictive controller exploits the multiparametric quadratic programming (mpQP) [18]. Therefore, any of the QP optimization problems

described in the Section 2.1 can be reformulated into their mpQP counterpart [16]:

$$\begin{aligned} \min_U \quad & \frac{1}{2}\, U^\top H U + \theta^\top F U && \text{(2.10a)} \\ \text{s.t.} \quad & G U \preceq E\theta + w, && \text{(2.10b)} \end{aligned}$$

where $\theta \in \mathbb{R}^n$ is the vector of parameters. The vector of the optimization variable $U \in \mathbb{R}^m$ is the vector of the manipulated variable optimized for the whole prediction horizon N, i.e., $U^\star = [u_0^{\star\top}, \ldots, u_{N-1}^{\star\top}]^\top$. Matrices $H \in \mathbb{R}^{m\times m} \succ 0$, $F \in \mathbb{R}^{n\times m}$, $G \in \mathbb{R}^{c\times m}$, $E \in \mathbb{R}^{c\times n}$, and vector $w \in \mathbb{R}^c$ define the problem data describing the system model and its limitations, and c represents the number of optimization problem constraints. Typically, the parameter θ defines the set of initial conditions of system states, for which the problem is solved in the offline phase [8].

The optimization problem (2.10) can be rewritten for

$$\begin{aligned} z &= U + H^{-1}F^\top\theta, && \text{(2.11a)} \\ S &= E + GH^{-1}F^\top, && \text{(2.11b)} \end{aligned}$$

into the equivalent following form [7]

$$\begin{aligned} \min_z \quad & 1/2\; z^\top H z && \text{(2.12a)} \\ \text{s.t.} \quad & G z \preceq S\theta + w. && \text{(2.12b)} \end{aligned}$$

Finally, the constraints of the optimization problem (2.12) can be divided as follows:

$$\begin{aligned} \min_z \quad & 1/2\; z^\top H z && \text{(2.13a)} \\ \text{s.t.} \quad & G_{\mathcal{A}} z = S_{\mathcal{A}}\theta + w_{\mathcal{A}}, && \text{(2.13b)} \\ & G_{\mathcal{N}} z \prec S_{\mathcal{N}}\theta + w_{\mathcal{N}}, && \text{(2.13c)} \end{aligned}$$

where $\mathcal{A}$ denotes the rows of matrices G, S, and vector w where the equality holds, i.e., the constraints are active. On the contrary, $\mathcal{N}$ denotes the inactive constraints. The index sets $\mathcal{A}$ and $\mathcal{N}$ are disjoint, i.e., $\mathcal{A} \cap \mathcal{N} = \varnothing$ and $\mathcal{A} \cup \mathcal{N} = \{1, \ldots, c\}$.

Let us consider a particular combination of $\mathcal{A}$ and $\mathcal{N}$. The corresponding Lagrange multipliers for the active constraints are denoted as $\lambda_{\mathcal{A}}$ and the Lagrange multipliers

of the inactive constraints are denoted as $\lambda_{\mathcal{N}}$. For (2.13), the Karush-Kuhn-Tucker (KKT) optimality conditions are [38]:

$$Hz^\star + G_{\mathcal{A}}^\top \lambda_{\mathcal{A}}^\star + G_{\mathcal{N}}^\top \lambda_{\mathcal{N}}^\star = 0, \tag{2.14a}$$
$$\lambda_{\mathcal{A}}^{\star\top}(G_{\mathcal{A}} z^\star - W_{\mathcal{A}} - S_{\mathcal{A}}\theta) = 0, \tag{2.14b}$$
$$\lambda_{\mathcal{N}}^{\star\top}(G_{\mathcal{N}} z^\star - W_{\mathcal{N}} - S_{\mathcal{N}}\theta) = 0, \tag{2.14c}$$
$$G_{\mathcal{A}} z^\star = W_{\mathcal{A}} + S_{\mathcal{A}}\theta, \tag{2.14d}$$
$$G_{\mathcal{N}} z^\star \prec W_{\mathcal{N}} + S_{\mathcal{N}}\theta, \tag{2.14e}$$
$$\lambda_{\mathcal{A}}^\star \succeq 0, \tag{2.14f}$$
$$\lambda_{\mathcal{N}}^\star = 0. \tag{2.14g}$$

As for all inactive constraints $G_{\mathcal{N}} z^\star - W_{\mathcal{N}} - S_{\mathcal{N}}\theta \prec 0$, from complementary slackness condition (2.14c) it follows that the corresponding Lagrange multipliers $\lambda_{\mathcal{N}}^\star = 0$. Then, from the stationarity condition (2.14a) we obtain affine relation between the optimization variable z and Lagrange multiplier $\lambda_{\mathcal{A}}$

$$z^\star = -H^{-1} G_{\mathcal{A}}^\top \lambda_{\mathcal{A}}^\star. \tag{2.15}$$

By substituting (2.15) into (2.14d) we get the following relation

$$\lambda_{\mathcal{A}}^\star = -(G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^\top)^{-1}(w_{\mathcal{A}} + S_{\mathcal{A}}\theta). \tag{2.16}$$

Finally, we substitute (2.16) into (2.15) and obtain

$$z^\star = H^{-1} G_{\mathcal{A}}^\top (G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^\top)^{-1}(w_{\mathcal{A}} + S_{\mathcal{A}}\theta). \tag{2.17}$$

To conclude, the result of the optimization problem in (2.13) is an affine relation between the optimization variable z and parameter θ:

$$z^\star = F(\mathcal{A})\theta + f(\mathcal{A}), \tag{2.18}$$

where the slope $F(\mathcal{A})$ and the section $f(\mathcal{A})$ depend on the specific combinations of the active and inactive constraints

$$F(\mathcal{A}) = H^{-1} G_{\mathcal{A}}^\top (G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^\top)^{-1} S_{\mathcal{A}}, \tag{2.19a}$$
$$f(\mathcal{A}) = H^{-1} G_{\mathcal{A}}^\top (G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^\top)^{-1} w_{\mathcal{A}}. \tag{2.19b}$$

The subset of the parametric space, where the affine control law in (2.18) is optimal, is defined as a critical region $\mathcal{R}$. The elements of the critical region $\mathcal{R}$ satisfy the primal feasibility (2.14e) and dual feasibility conditions (2.14f) [5] and result in a closed and bounded polytope

$$\mathcal{R} = \left\{ \theta \in \mathbb{R}^n \mid A\theta \preceq b \right\}, \tag{2.20}$$

where

$$A = \begin{bmatrix} G_{\mathcal{N}} F(\mathcal{A}) - S_{\mathcal{N}} \\ (G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^{\top})^{-1} S_{\mathcal{A}} \end{bmatrix}, \tag{2.21a}$$

$$b = \begin{bmatrix} w_{\mathcal{N}} - G_{\mathcal{N}} f(\mathcal{A}) \\ -(G_{\mathcal{A}} H^{-1} G_{\mathcal{A}}^{\top})^{-1} w_{\mathcal{A}} \end{bmatrix}. \tag{2.21b}$$

Note, the closure of the critical region is obtained by replacing the strict inequalities with the non-strict ones.

We can see in (2.19) and (2.21), that every feasible combination of the active and inactive constraints defines a specific critical region $\mathcal{R}$ with its corresponding affine control law in (2.18). Therefore, when the multiparametric QP in (2.13) is solved for the whole parametric space, one can obtain the complete piecewise affine control law defined over all regions, i.e., over the polytopic partition given by $\cup_i^{R_{\text{total}}} \mathcal{R}_i$, where R_{total} denotes the total number of the generated critical regions, see Figure 2.4.

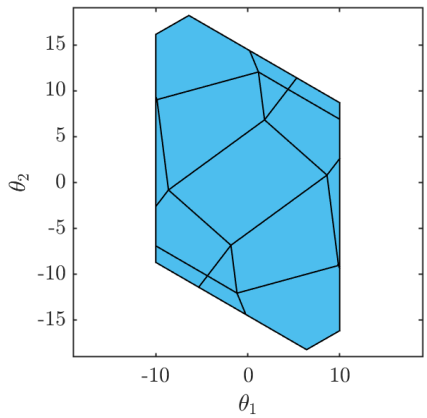


Figure 2.3: Example of polytopic partition.

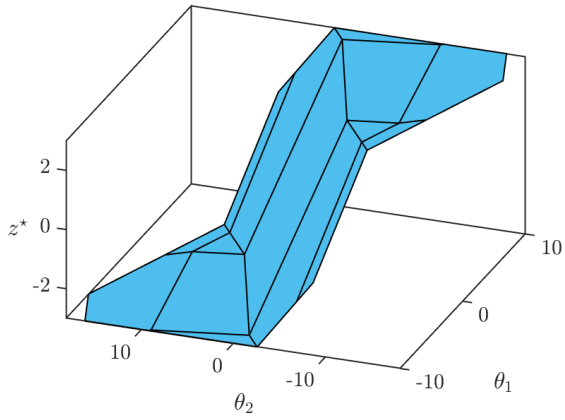


Figure 2.4: Example of piecewise affine control law.

CR_1	$AS_1 = \{1, \dots, k_0, k_1\}$ *or* $AS_1 = \{1, \dots, k_0\} \backslash k_1, k_1 \in \{1, \dots, k_0\}$
CR_0	$AS_0 = \{1, \dots, k_0\}$
CR_2	$AS_2 = \{1, \dots, k_0, k_2\}$ *or* $AS_2 = \{1, \dots, k_0\} \backslash k_2, k_2 \in \{1, \dots, k_0\}$

Figure 2.5: Optimal active sets of adjacent regions [32].

Definition 2.2.1. ***(Adjacent critical regions)*** *[78] Given a full-dimensional critical region $\mathcal{R}_i$. A critical region $\mathcal{R}_j$ is adjacent to $\mathcal{R}_i$ if $int(\mathcal{R}_i) \cap int(\mathcal{R}_j) = \varnothing$ and $\mathcal{R}_i \cap \mathcal{R}_j \neq \varnothing$.*

Definition 2.2.2. ***(Optimal active sets of adjacent critical regions)*** *[81] For any two adjacent critical regions $\mathcal{R}_i$ and $\mathcal{R}_j$ the corresponding optimal active sets are $\mathcal{A}_i \subset \mathcal{A}_j$ and $|\mathcal{A}_i| = |\mathcal{A}_j| - 1$ or $\mathcal{A}_j \subset \mathcal{A}_i$ and $|\mathcal{A}_i| = |\mathcal{A}_j| + 1$, see Figure 2.5.*

Definition 2.2.3. ***(Linear independence constraint qualification (LICQ))*** *[8] Linear independence constraint qualification is said to hold at $z^\star$ if the matrix $G_\mathcal{A}$ has full row rank.*

Once the explicit model predictive controller is constructed, it can be utilized for control. In the online phase of standard explicit MPC implementation, the point location problem is solved. According to system state measurement θ, the corresponding critical region is located such that $\theta \in \mathcal{R}_i$ holds. When the corresponding critical region is detected, the associated affine control law is utilized to implement the optimal control action $u_0^\star$ into the system. In every control step, the whole procedure is repeated.

2.2.1 Construction of explicit MPC

In order to obtain the explicit PWA solution for (2.12), the critical regions $\mathcal{R}_i$ need to be identified along with the corresponding optimizer $z_i^\star$ for each region. There are two types of methods that can be used to obtain the list of optimal active sets: geometric approaches and enumeration procedures.

- **Geometric approaches**
  One common approach to solving the mpQP problem is the geometric method, which involves constructing an initial critical region by selecting a feasible parameter value θ. This critical region provides information about the optimal active set $\mathcal{A}$ by solving the mpQP in (2.12). Different geometric methods vary in how they explore the rest of the parameter space. For instance, some methods, such as those presented in [5] and [16], employ set difference operations to explore the parameter space based on the known critical regions. Other methods, like the one proposed in [81], select a new point by traversing the facets of the known critical region. Obviously, all methods are employed until the entire parameter space is covered.

- **Enumeration procedures**
  In contrast to the geometric methods, the extensive enumeration approach generates optimal active sets without the necessity to construct the critical regions [21]. First, all possible combinations of active sets are enumerated and organized by the increasing cardinality. To determine whether a particular

candidate is optimal, the following KKT-system-based linear program is solved:

$$\begin{aligned}
\max_{g,z,\theta,\lambda} \quad & g && \text{(2.22a)}\\
\text{s.t.} \quad & Hz + G_{\mathcal{A}}^{\top}\lambda = 0, && \text{(2.22b)}\\
& G_{\mathcal{A}} z = S_{\mathcal{A}}\theta + w_{\mathcal{A}}, && \text{(2.22c)}\\
& g \preceq S_{\mathcal{N}}\theta + w_{\mathcal{N}} - G_{\mathcal{N}} z, && \text{(2.22d)}\\
& \lambda \succeq 0, && \text{(2.22e)}\\
& g \succeq 0. && \text{(2.22f)}
\end{aligned}$$

If the problem (2.22) is feasible with $g^{\star} \succeq 0$, the active set candidate is optimal and yields a full-dimensional critical region. On the contrary, if the linear program is infeasible, the candidate together with all the candidates in the corresponding branch are not considered, i.e., pruned. This pruning leads to significant complexity reduction of the offline phase. This idea is extended in [42] where the authors suggest not storing the polytopic representation of the critical regions. Instead, only analytical expressions of primal and dual variables are stored, which leads to significant memory footprint reduction.

With increasing problem size, i.e., length of prediction horizon N, parameter size n or size of the optimization variable m, the overall complexity of the solution is also rising. Therefore, the number of critical regions is large. Then, the memory burden and point location problem become more complex, and the usage of the explicit solution is hardly tractable.

2.2.2 Degeneracies in mpQP

In practical applications, degeneracies in mpQP problem solutions may occur. The primal degeneracy occurs if linear independence constraint qualification (LICQ) does not hold, i.e., the rows of matrix $G_{\mathcal{A}}$ are not linearly independent, see e.g. [81]. In this case, the vector of Lagrange multipliers $\lambda^{\star}$ may not be uniquely defined. As a consequence, it leads to the presence of the lower-dimensional critical regions in the explicit solution. These lower-dimensional critical regions can be excluded from the solution, as they form the facets between neighboring full-dimensional critical regions [5].

The dual degeneracy [5] occurs when the objective function is linearly dependent on some active constraint and leads to nonunique $z^{\star}$. In the problem formulation in this thesis, the dual degeneracy cannot occur, as the objective function is strictly quadratic,

i.e., $H \succ 0$, see problem formulation in (2.10) and the corresponding assumptions below.

2.3 Partial explicit MPC

One of the perspective techniques on how to handle large-scale optimization problems with numerous active sets is a partial solution of the explicit MPC, see [32]. The main idea is to solve the problem of explicit MPC only for particular initial points from the feasible domain, i.e., feasible seeding points. Therefore, only a subset of all critical regions is determined and stored, i.e., $\mathcal{R}_i$, $i = 1, \ldots, R$, where R denotes the number of the evaluated and stored critical regions. The partial solution of explicit MPC can be seen in Figure 2.6.

Since not all critical regions are constructed, it can often occur in the online phase that the current state measurement θ does not lay inside any of them, i.e., $\theta \notin \mathcal{R}_i$, $\forall i = 1, \ldots, R$. In such cases, the optimal solution of the optimization problem in (2.12) needs to be solved to find the optimal control action.

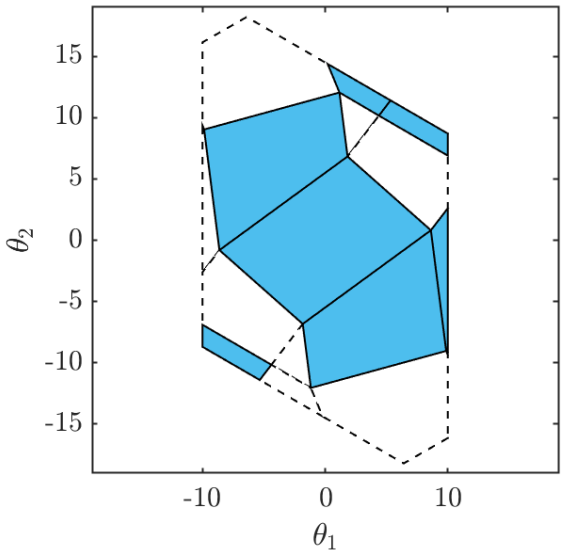


Figure 2.6: Example of partial solution of explicit MPC. The blue critical regions are constructed and stored for the online phase.

To initialize evaluation of the optimal control action, a near critical region is utilized. This is a useful tool to streamline searching for the critical region where the measurement belongs. This procedure is called hot start strategy [17]. Once the critical region or associated active set is identified, then the corresponding control law is determined to find the optimal control action.

2.3.1 Initial seeding points

There are various ways of determining the set of initial seeding points, for which the QP in (2.12) is solved in the offline phase. One of the ways is to generate a grid of points uniformly distributed in the feasible parameter space. Another way is to distribute the seeding points randomly, utilizing so-called random walks. The random walks represent an efficient strategy to obtain a random distribution of points inside a large-scale polytope (parameter space), as the aim is to develop mainly those critical regions that are volumetrically significant and not to explore the feasible parameter space entirely. Various random walk techniques exist in the literature, e.g. [12] and [76].

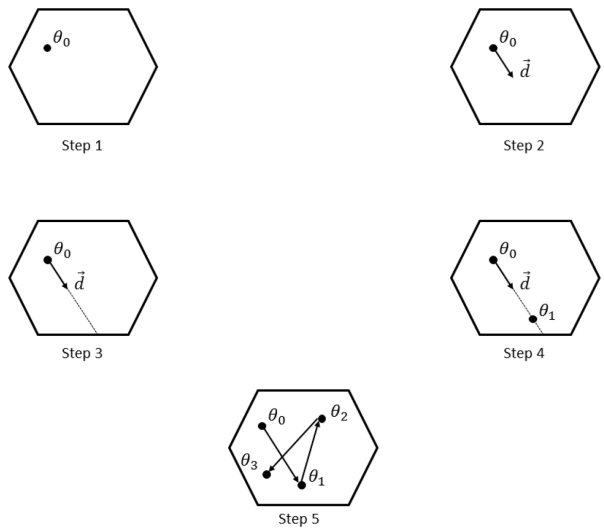


Figure 2.7: Hit and run sampling principle [32].

In [32], the authors follow up on work [76] denoted as hit and run sampling. The main steps are as follows. The starting point θ_0 to initialize the algorithm is selected, and a random direction $\vec{d}$ is determined. Then, the maximum distance, denoted as s, the parameter θ_0 can move along the direction $\vec{d}$ staying feasible is identified from:

$$\begin{aligned} \max_s \quad & s && \text{(2.23a)} \\ \text{s.t.} \quad & GU \preceq E(\theta + \vec{d}s) + w, && \text{(2.23b)} \\ & s > 0. && \text{(2.23c)} \end{aligned}$$

After that, a random value between $(0, s\rangle$ is determined, and a new point θ_1 is identified by moving along the direction vector $\vec{d}$ by this amount. The procedure is then repeated until enough seeding points are collected. The principle of the hit and run sampling procedure can be seen in Figure 2.7.

By increasing the number of sample points, it is possible to identify more critical regions defining the partial explicit solution. On the other hand, more critical regions have to be stored, which leads to an increased memory burden. Therefore a balance in determining a sufficient number of sample points is needed.

2.3.2 Hot start strategy

In the online phase of the partial explicit MPC, the hot start strategy is utilized to streamline the identification of the critical region where the current value of the parameter lies. The hot start strategy utilizes a near parameter realization θ_i, its corresponding region $\mathcal{R}_i$, and the current parameter θ_j, to find the critical region $\mathcal{R}_j$ such that $\theta_j \in \mathcal{R}_j$ [17]. The benefit of the hot start technique lies in exploiting the property of two adjacent critical regions, see Definition 2.2.2. As the optimal active sets of the neighboring critical regions differ in only one constraint, this constraint is identified, and the neighboring critical region is determined.

The procedure is summarized as follows. Given a measurement θ_j and a near value of parameter θ_i along with its corresponding critical region $\mathcal{R}_i$. First, it is checked whether the measurement $\theta_j \in \mathcal{R}_i$. If it lies in the critical region $\mathcal{R}_i$, the corresponding control law can be directly utilized to determine the optimal control action. If not, a direction vector $\vec{d}$ from θ_i to θ_j is identified. The next step is to find the intersection of the vector $\vec{d}$ and the facet of $\mathcal{R}_i$. In other words, a constraint defining the critical

region that will be violated first is identified:

$$l = \arg\min_{l} \ \gamma_l \geq 0, \quad l = 1, \ldots, |b|, \tag{2.24}$$

where

$$\gamma = (b - A\theta_i) \oslash A\vec{d}, \tag{2.25}$$

with the symbol $\oslash$ representing the Hadamard division [80]. The intersected facet corresponds to the index l associated with minimal non-negative $\gamma^\star$. This constraint is either added or removed from the active set combination defining $\mathcal{R}_i$.

The determination of the optimal active set of the adjacent critical region is based on Definition 2.2.2. Given the optimal active set combination $\{i_1, \ldots, i_k\}$ and its corresponding critical region $\mathcal{R}_0$. Let $\mathcal{R}_1$ be a neighboring critical region, and $\mathcal{H}$ denotes the separating hyperplane. The vector λ represents the Lagrange multipliers associated with the active set combination of $\mathcal{R}_0$.

- If $\mathcal{H}$ is defined by $G_{i,k+1}U = E_{i,k+1}\theta + w_{i,k+1}$, then $\mathcal{R}_1$ is defined by the active set combination $\{i_1, \ldots, i_k, i_{k+1}\}$.

- If $\mathcal{H}$ is defined by $\lambda_{i,k} = 0$, then $\mathcal{R}_1$ is defined by the active set combination $\{i_1, \ldots, i_{k-1}\}$.

Given the new active set, the adjacent critical region is found. After that, the parameter value θ_j is checked to exist in this new critical region, and the procedure is repeated until the associated critical region $\mathcal{R}_j$ is found. The principle of the hot start strategy is depicted in Figure 2.8.

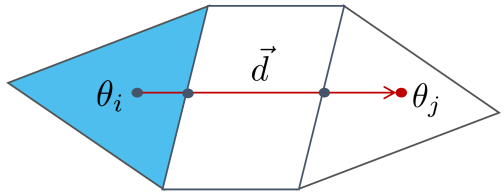


Figure 2.8: Principle of hot start strategy utilized for identifying the critical region associated with parameter θ_j. The blue critical region is known along with its internal point θ_i. The grey points are the intersections of the facets of critical regions and the direction vector $\vec{d}$ between the two parameter values.

Implementation of the hot start strategy requires storing one point, which lies inside every stored critical region, a so-called feasible point. In [32], it is suggested to construct the Chebyshev ball inside each considered critical region and store its center.

2.3.3 Chebyshev ball

In this section, the problem of finding the Chebyshev ball is described. The Chebyshev ball represents the maximal volume inner approximation of a polytope $\mathcal{R}$ described by h linear inequalities [9]

$$\mathcal{R} = \{x \in \mathbb{R}^n | \alpha_i^\top x \preceq \beta_i, i = 1, \dots, h\}, \tag{2.26}$$

where α and β describe the halfspaces of the polytope.

The Chebyshev ball is identified from the linear program [9]

$$\begin{aligned} \max_{r,C} \quad & r && \text{(2.27a)} \\ \text{s.t.} \quad & \alpha_i^\top C + r\|\alpha_i\|_2 \preceq \beta_i, \quad i = 1, \dots, h, && \text{(2.27b)} \end{aligned}$$

where the optimization variables are the radius r and the center C of the Chebyshev ball.

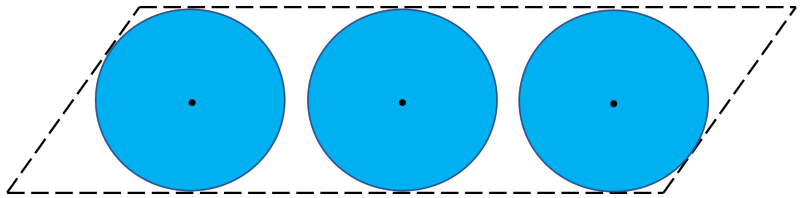


Figure 2.9: Example of non-unique Chebyshev balls.

The Chebyshev balls' centers C_i are then stored along with the associated critical regions $\mathcal{R}_i$, $\forall i = 1, \ldots, R$. It should be noted that problem 2.27 may have multiple optimal solutions, i.e., the Chebyshev ball inscribed in the given polytopic region $\mathcal{R}_i$ is not unique in general, see Figure 2.9. Multiple maximal volume balls can be inscribed in the polytope, e.g., when the critical region has a shape of hyperbox, trapezoid, parallelotope, etc. Nevertheless, any Chebyshev ball can be considered, as the aim is to find a feasible point laying inside the specific critical region.

2.3.4 Procedure

The procedure of the partial explicit MPC based on [32] is described in the two following algorithms, where Algorithm 1 summarizes the steps of the offline phase, and Algorithm 2 summarizes the steps of the online phase.

Algorithm 1: Offline phase of partial explicit MPC [32].

Inputs: Set of R random feasible seeding points $\{p_1, p_2, \ldots p_R\}$
Outputs: Matrices A_i and b_i of critical regions $\mathcal{R}_i$ in (2.20), centers of Chebyshev balls C_i, binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$

1: **for each** p_i **do:**
2: solve the QP in (2.12) for $\theta \leftarrow p_i$
2: find and store optimal active set $\mathcal{I}_i$
3: construct the critical region $\mathcal{R}_i$ in (2.21)
4: store the polytope matrices A_i, b_i
5: **for each** $\mathcal{R}_i$ **do:**
6: construct Chebyshev ball $\widetilde{\mathcal{R}}_i$
7: store Chebyshev ball center C_i of $\widetilde{\mathcal{R}}_i$

Algorithm 2: Online phase of partial explicit MPC [32].

Inputs: Parameter value θ, matrices A_i and b_i of critical regions $\mathcal{R}_i$, centers of Chebyshev balls C_i, binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$, optimization problem matrices H, G, w, S in (2.12)
Output: Optimal control action $u_0^\star$

1: **for each** $\mathcal{R}_i$ **do:**
2: $\quad v_i = \|\theta - C_i\|$
3: solve $v^\star = \min v_i, \quad \forall v_i.$
4: find the region $\mathcal{R}_i$ and associated active set $\mathcal{I}_i$ corresponding to the minimal $v^\star$
5: **if** $\theta \in \mathcal{R}_i$:
6: $\quad$ apply the optimal control action $u_0^\star$ using $\mathcal{I}_i$ and (2.18)
7: **else**:
8: $\quad$ find optimal active set $\mathcal{I}^\star$ from hot start procedure using $\mathcal{I}_i$
9: $\quad$ apply the optimal control action $u_0^\star$ using $\mathcal{I}^\star$ and (2.18)

In the offline phase, the QP in (2.12) is solved for every feasible seeding point p_i. The solution in the form of optimal active sets is then stored for online phase as a binary vector $\mathcal{I}_i$. This vector has a cardinality given by the number of constraints, i.e., $|G|$ in (2.12). The elements of the binary vector represent the fixed-ordered indices of constraints, and their binary values indicate if the particular constraint is active or inactive, e.g., see [61]. When the optimal active set is available, the corresponding critical region is constructed and the matrices A and b defining the polytope are stored. The last steps of the offline phase are dedicated to finding and storing the Chebyshev ball center and radius for every critical region.

Remark 2.3.1. *(Storing the control law) As binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$, defining the active sets are stored, the matrices $F(\mathcal{A})$ and $f(\mathcal{A})$ are recovered from the problem matrices H, G, w, and S to apply the optimal control action $u_0^\star$. It is also possible to store the matrices $F(\mathcal{A})$ and $f(\mathcal{A})$, along with the polytopic regions $\mathcal{R}_i$, but it leads to an unnecessary increased memory burden.*

Remark 2.3.2. *(Number of critical regions) The random distribution of the feasible seeding points p_i, $i = 1, \ldots, R$, that serve to initialize the partial parametric solution of the optimization problem in the offline phase could lead to multiple evaluations of the same critical region, i.e., $p_i \mapsto \mathcal{R}_i$, $p_j \mapsto \mathcal{R}_i$, for $p_i \neq p_j$. In such case, it is sufficient to store such a critical region just once. Then two options are: (i) accept the number of the unique critical regions lower than the number of the feasible seeding points, or (ii) insert a new random feasible seeding point until the required number of the unique critical regions is evaluated.*

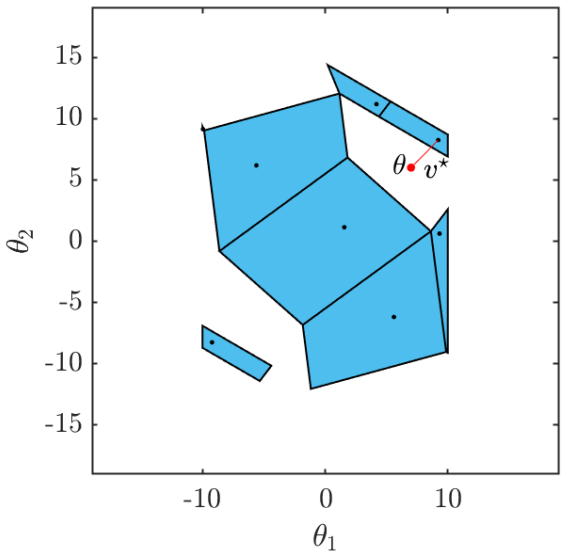


Figure 2.10: Example of partial explicit MPC: determination of the critical region utilized in the hot start strategy. The critical region associated with the minimal distance $v^\star$ (red line) between its Chebyshev center and the parameter value θ is exploited to find the optimal control action.

In the online phase, the partial explicit solution is exploited. First, a distance between the current parameter and every Chebyshev center is evaluated. The optimal active set of the critical region associated with the nearest Chebyshev center is identified, see Figure 2.10. Afterward, it is checked whether the parameter lies in the critical region. If the parameter lies in the polytope, the corresponding optimal active set is utilized to apply the optimal control action. Otherwise, the hot start procedure is performed to find the optimal active set corresponding to the current parameter value. In the next control step, the whole procedure of Algorithm 2.3.4 is performed again.

2.3.5 Memory footprint

For large-scale optimization problems, it is essential to investigate the memory burden. In the online phase, the controller utilizes the data H, G, w, and S of the optimization problem in (2.12), as well as the partial solution represented by matrices A_i, b_i

defining the critical regions $\mathcal{R}_i$ in (2.20), see inputs of Algorithm 2. Moreover, the feasible points of all stored regions, i.e., the Chebyshev centers C_i, $i = 1, \ldots, R$, are saved. The problem size is given by the parameter dimension n, dimension of the optimization variable m, and the length of the prediction horizon N. Obviously, if the problem size is large, then the demands on memory storage become hardly tractable.

Lemma 2.3.3 (Memory footprint of polytopic region)**.** *Given mpQP in* (2.12)*. The memory footprint of the i-th critical region $\mathcal{R}_i \subset \mathbb{R}^n$ having a form of a polytope given by h_i halfspaces defined in* (2.20) *requires following number of floating-point numbers necessary to store this polytope*

$$n(\mathcal{R}_i) = h_i \times n + h_i. \tag{2.28}$$

Proof: Proof of Lemma 2.3.3 directly follows from the structure of the polytopic i-th critical region $\mathcal{R}_i$, i.e., the data necessary to store h_i halfspaces determined by a matrix $A_i \in \mathbb{R}^{h_i \times n}$ and vector $b_i \in \mathbb{R}^{h_i}$ in (2.20). □

The memory footprint necessary to store the data of partial solution is not only large but it is also unpredictable. Although the number of the critical regions is determined by the number of random points R, and the upper bound on the data size defining the critical region is known, it is not possible to predict the exact size of the data that we need to store in advance before solving the optimization problem. It is obvious from Lemma 2.3.3, that the number of floating-point numbers $n(\mathcal{R}_i)$ in (2.28) necessary to store the critical region $\mathcal{R}_i$ is not fixed and varies from one region to another. Particularly, the value of $n(\mathcal{R}_i)$ depends on the number of halfspaces h_i that define the i-th specific critical region.

Theorem 2.3.4 (Memory footprint of polytopic partial solution)**.** *Given mpQP in* (2.12)*. The total memory footprint of partial solution consisting of R critical regions including the optimization problem matrices H, G, S, w, and R binary vectors of active constraints $\mathcal{I}$, requires following number of floating point numbers*

$$n(\forall\mathcal{R}) \quad = \quad \sum_{i=1}^{R} (n(\mathcal{R}_i)) + R\,n + R\frac{c}{64} + (m \times m + c \times m + c \times n + c), \tag{2.29}$$

where n is the parameter dimension, m is number of the optimization variables, c represents the number of constraints, and $n(\mathcal{R}_i)$ denotes the memory footprint of the i-th specific critical region in (2.28)*.*

Proof: Proof of Theorem 2.3.4 directly follows from the structure of the polytope in (2.28), dimensions of the optimization problem matrices in (2.12), and the number of the critical regions R. According to Lemma 2.3.3, the memory consumption of the i-th polytopic critical regions $n(\mathcal{R})$ is defined by finite number of halfspaces h_i determined by a matrix $A_i \in \mathbb{R}^{h_i \times n}$ and vector $b_i \in \mathbb{R}^{h_i}$ in (2.20). The remaining data contains: memory footprint of corresponding binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$ and matrices of the optimization problem H, G, S, w in (2.12). Note, the memory footprint necessary to store the binary vector $\mathcal{I}$ defining the active sets is divided by 64 to transform the binary format to the double floating point numbers. □

According to Theorem 2.3.4, it is necessary to solve the large-scale optimization problem to determine the total memory footprint.

2.4 Fixed-memory partial explicit MPC design

This section is devoted to one of the main contributions of this thesis, i.e., designing the fixed-memory parametric solution of partial explicit MPC. The term "fixed-memory" denotes that the size of the memory footprint necessary to store the parametric solution is determined in advance, i.e., before solving the multiparametric optimization problem in (2.12). This is a groundbreaking benefit enabling us to scale the solution of the partial explicit MPC to respect the limited memory of the hardware, where the controller will be installed.

We recall that in the MPC framework introduced in [32], the partial solution of explicit MPC is evaluated. The critical regions are stored in the form of polytopes $\mathcal{R}_i$ in (2.20), and also the Chebyshev balls' centers C_i are stored, see Algorithm 1. The centers of Chebyshev balls C_i are then used in the online phase to evaluate the distance to the measurement θ. The Chebyshev ball center nearest to the measurement θ is determined and the corresponding critical region $\mathcal{R}_i$ is used for initialization of the hot start procedure to solve the large-scale optimization problem in (2.12) for a given state measurement θ, see Algorithm 2.

The main idea of a fixed memory footprint is to replace the polytopic region $\mathcal{R}_i$ with its maximal volume inner approximation $\widetilde{\mathcal{R}}_i$ using the Chebyshev ball. In other words, it is not necessary to store the large data A_i, b_i defining the polytopes $\mathcal{R}_i$ in (2.20), but only the light-weight data defining the Chebyshev balls $\widetilde{\mathcal{R}}_i$ are stored. As a consequence, the memory footprint of each critical region is fixed.

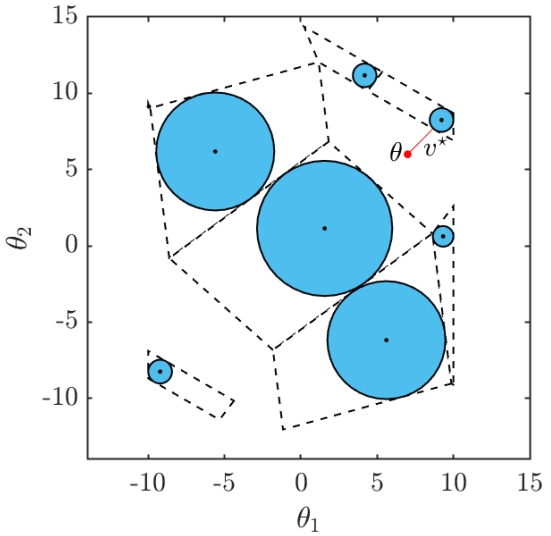


Figure 2.11: Example of fixed-memory partial solution of explicit MPC. The polytopic regions $\mathcal{R}_i$ (black dashed) are not stored. Instead, the Chebyshev ball approximations $\widetilde{\mathcal{R}}_i$ are stored (blue). In the online phase, the minimum distance $v^\star$ from the set of Chebyshev balls (red line) to the current measurement θ is found.

Except for fixing the memory footprint, this approach leads to significant memory savings. Compared to storing the polytopic representation of the critical region $\mathcal{R}_i$, the memory savings are ensured, as just a single point (center of Chebyshev ball C_i) and a scalar (radius of Chebyshev ball r_i) are stored for each critical region, $\widetilde{\mathcal{R}}_i$, $\forall i = 1, \ldots, R$. The principle of fixed-memory partial solution of explicit MPC can be seen in Figure 2.11.

Moreover, another benefit of this approach is that the evaluation of the nearest critical region to the current state measurement θ is more accurate compared to the one introduced in [32].

In [32], the distance v_i between the current system measurement θ and the critical region $\mathcal{R}_i$ is evaluated by the center of the Chebyshev ball C_i, i.e., $v_i = \|\theta - C_i\|$. In contrast, the novel approach evaluates the distance v_i using the boundary of Chebyshev ball given by its radii r_i, i.e., $v_i = \|\theta - C_i\| - r_i$. Therefore, the distance v_i between

the current system measurement θ and the original polytopic set $\mathcal{R}_i$ approximated by the Chebyshev ball $\widetilde{\mathcal{R}}_i$ is more accurate in comparison to the original approach based just on a center of inscribed Chebyshev ball C_i, see Figure 2.12.

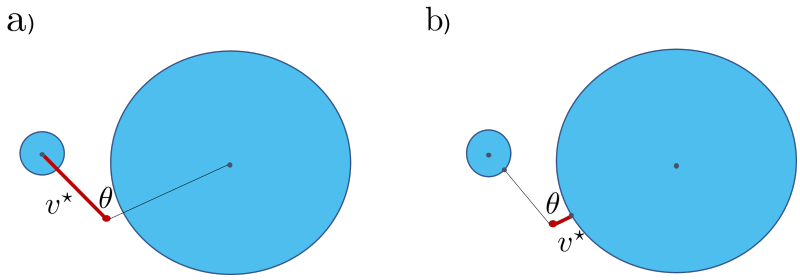


Figure 2.12: Comparison of determination of the nearest critical region. In a), the idea is adapted from [32], where the critical region with the nearest Chebyshev ball's center was utilized for the hot start strategy. In b), the novel approach is employed, i.e., the critical region with the nearest Chebyshev ball is determined and utilized, which leads to more accurate initialization of the hot start strategy.

2.4.1 Procedure

The Algorithm 1 and Algorithm 2 were revisited to demonstrate the online phase and offline phase of the fixed-memory partial explicit MPC procedure.

The offline phase procedure of fixed-memory partial explicit MPC is evaluated by Algorithm 3. Compared to the original procedure in Algorithm 1, the novel approach is extended by storing the Chebyshev ball radii r_i. On the other hand, the polytopic representations of the critical regions A_i, b_i in (2.20) are not stored. If necessary, A_i, b_i can be recovered in the online phase from the binary vector $\mathcal{I}_i$ defining the corresponding active set $\mathcal{A}$ and the matrices H, G, S, w of the optimization problem in (2.12).

The online phase of the fixed-memory partial explicit MPC is described in Algorithm 4, where the distances from the boundaries of the Chebyshev balls are identified (Algorithm 4, Step 2), in contrast to the original approach in [32], cf. Algorithm 2, Step 2.

In Algorithm 2, if current system measurement θ lies inside some of the approximated critical regions $\widetilde{\mathcal{R}}_i$, then the optimal control action is evaluated using the corresponding

control law in (2.18). Otherwise, if θ does not lie inside any of the Chebyshev balls, i.e., $\theta \notin \widetilde{\mathcal{R}}_i$, $\forall i = 1, \ldots, R$, then the nearest Chebyshev ball is used for hot started solution of the large-scale optimization problem in (2.12). The rest of the procedure remains the same as proposed in [32].

Algorithm 3: Offline phase of fixed-memory partial explicit MPC.

Input: Set of R random feasible points $\{p_1, p_2, \ldots p_R\}$
Outputs: Centers of Chebyshev balls C_i, radii of Chebyshev balls r_i, binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$

for each p_i **do:**
    solve the QP in (2.12) for $\theta \leftarrow p_i$
    find and store optimal active set binary vector $\mathcal{I}_i$
    construct the critical region $\mathcal{R}_i$ in (2.21)
for each $\mathcal{R}_i$ **do:**
    construct Chebyshev ball $\widetilde{\mathcal{R}}_i$
    store center C_i and radius r_i of $\widetilde{\mathcal{R}}_i$

Algorithm 4: Online phase of fixed-memory partial explicit MPC.

Inputs: Parameter value θ, centers of Chebyshev balls C_i, radii of Chebyshev balls r_i, binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$, optimization problem matrices H, G, S, w
Output: Optimal control action $u_0^\star$

for each C_i **do:**
    $v_i = \|\theta - C_i\| - r_i$
    if $(v_i < r_i)$:
        apply the optimal control action $u_0^\star$ using $\mathcal{I}_i$
        break
solve $v^\star = \min v_i$, $\forall v_i$
identify active set $\mathcal{I}_i$ of the nearest region for $v^\star$
construct the polytopic critical region $\mathcal{R}_i$ using $\mathcal{I}_i$ and optimization problem matrices H, G, S, w
if $\theta \in \mathcal{R}_i$:
    apply the optimal control action $u_0^\star$ using $\mathcal{I}_i$
else:
    find optimal active set $\mathcal{I}^\star$ from hot start procedure using $\mathcal{I}_i$
    apply the optimal control action $u_0^\star$ using $\mathcal{I}^\star$ and (2.18)

Note, the proposed approach interferes with the online phase of the original approach

adapted from [32] in one major step – construction of the nearest polytope, see Algorithm 4, Step 8. After the nearest critical region is estimated, the rest of the online procedure remains the same. The remainder of the modifications represents a negligible computational intervention compared to the construction of the polytope, i.e., checking whether the parameter lies in the Chebyshev ball in Algorithm 4, Step 3, and subtracting the radius in Algorithm 4, Step 2.

Remark 2.4.1 (Degeneracies in mpQP)**.** *In practical applications, primal degeneracy in mpQP problem solutions may occur, see Section 2.2.2. It leads to the presence of the lower-dimensional critical regions in the (partial) explicit solution. In the offline phase, these lower-dimensional critical regions can be excluded from the solution, as they form the facets between neighboring full-dimensional critical regions. Then, during the hot start strategy in the online phase, it is checked whether LICQ holds on the facet of a polytope intersecting the direction vector. If LICQ does not hold, a QP is solved with improved initialization based on the known critical region.*

2.4.2 Memory footprint

To support the benefits of the proposed ideas, the memory footprint associated with the fixed-memory approach was investigated as well as the original approach in [32].

Lemma 2.4.2 (Memory footprint of approximated region)**.** *Given mpQP in* (2.12)*. The memory footprint of the approximation of the i-th critical region* $\widetilde{\mathcal{R}}_i \subset \mathbb{R}^n$ *using the Chebyshev ball is fixed and requires following number of floating-point numbers*

$$n(\widetilde{\mathcal{R}}) = 1 + n. \tag{2.30}$$

Proof: Proof of Lemma 2.4.2 directly follows from the fixed structure of the Chebyshev ball, i.e., the data necessary to store the Chebyshev balls' radius $r \in \mathbb{R}$ and coordinates of its center $C \in \mathbb{R}^n$. □

Remark 2.4.3. *The fixed memory footprint of* $\widetilde{\mathcal{R}}_i$ *could be ensured also by other well-known maximal volume inner approximations of the polytopic critical region* $\mathcal{R}_i$ *in* (2.20)*, e.g., by hyperboxes or ellipsoids. In this thesis, the Chebyshev balls are considered as they lead to a reasonable trade-off between the numerical complexity and the volume of the approximation* $\widetilde{\mathcal{R}}_i$*. Obviously, introducing inner approximation may lead to a situation when the critical region is not detected in the online phase, although the critical region is known, i.e.,* $\theta \in \widetilde{\mathcal{R}}_i \Rightarrow \theta \in \mathcal{R}_i$*, but* $\theta \in \mathcal{R}_i \not\Rightarrow \theta \in \widetilde{\mathcal{R}}_i$*.*

As the inner approximation $\widetilde{\mathcal{R}}_i$ using the Chebyshev ball of each critical region $\mathcal{R}_i$ has the same fixed structure $\forall i = 1, \ldots, R$, the fixed memory footprint of each region is

enforced, see Lemma 2.4.2. Next, the fixed-size memory footprint necessary to store all data utilized in the online phase is determined.

Theorem 2.4.4 (Memory footprint of approximated solution)**.** *Given mpQP in* (2.12)*. The total memory footprint of partial solution consisting of R approximated critical regions, including the optimization problem matrices H, G, S, w, and R binary vectors indicating the set of active constraints $\mathcal{I}_i$ requires following number of floating point numbers*

$$n(\forall\widetilde{\mathcal{R}}) = R\,n(\widetilde{\mathcal{R}}) + R\frac{c}{64} + (m \times m + c \times m + c \times n + c). \tag{2.31}$$

where $n(\widetilde{\mathcal{R}})$ denotes the memory footprint of one critical region approximated using the Chebyshev ball in (2.30)*.*

Proof: Proof of Theorem 2.4.4 directly follows from the fixed structure of the Chebyshev ball, dimensions of the optimization problem matrices, and the number of the critical regions R. According to Lemma 2.4.2, the memory consumption of the approximated critical regions $n(\widetilde{\mathcal{R}})$ is defined by Chebyshev balls' radii r_i and centers C_i. The remaining data contains: memory footprint of corresponding binary vectors $\mathcal{I}_i$, $i = 1, \ldots, R$ and matrices of the optimization problem H, G, S, w in (2.12). We recall, the memory footprint necessary to store the binary vector $\mathcal{I}$ defining the active sets is divided by 64 to transform the binary format to the double floating point numbers. □

Corollary 2.4.4.1. *The total memory footprint of a partial solution $n(\forall\widetilde{\mathcal{R}})$ in Theorem* (2.4.4) *is an affine function of the number of the considered approximated critical regions R, and is independent on the solution of the optimization problem in* (2.12)*.*

Proof: First, let us prove that (2.31) is an affine function of R. According to Theorem 2.4.4, the total memory footprint of partial solution is given by (2.31), that can be rewritten into an equivalent form of a affine function of R given by:

$$n(\forall\widetilde{\mathcal{R}}) = p\,R + q, \tag{2.32}$$

where the slope is $p = (n(\widetilde{\mathcal{R}}) + \frac{c}{64})$ and the section is $q = (m \times m + c \times m + c \times n + c)$. Next, it is shown that (2.32) is independent on the solution of the optimization problem in (2.12). Function (2.32) is defined by p, q, and depends on variable R. The parameters p, q are determined only by the size of the optimization problem in (2.12), and the number of feasible points R is given in advance. Therefore, p, q are evaluated without the necessity to solve the optimization problem in (2.12). □

2.5 Numerical results

This section is devoted to the demonstration of the benefits of fixed-memory partial explicit MPC. We recall that the proposed approach interferes with the online phase of the original approach adapted from [32] in one major step – construction of the nearest polytope, see Algorithm 4, Step 8. The remaining modifications were minor, i.e., checking whether the parameter lies in the Chebyshev ball in Algorithm 4, Step 3, and subtracting the radius in Algorithm 4, Step 2. Therefore, the proposed operations still represent a negligible part compared to the operations in the rest of the control action evaluation procedure, e.g., the hot start strategy where multiple critical regions are constructed. Because of this and the fact that the optimality of control input is not affected by the proposed improvements, only the offline phase is analyzed. The aim of this section is to analyze the memory footprint of the original approach based on the polytopic representation of critical regions presented in [32] and the proposed fixed-memory approach based on the maximal volume inner approximation using the Chebyshev ball.

First, 5 differently large sets of 5 random large-scale systems were generated. For every system model, the mpQP problem was formulated and partial solution was constructed based on both approaches. For both approaches, the average memory footprint of the solution was evaluated and compared. Finally, the memory savings associated with the proposed approach were evaluated.

The advantages and properties of the proposed fixed-memory approach were demonstrated and analyzed on the memory footprint comparison. First, the memory footprint of the partial explicit solution was evaluated based on the original approach exploiting the polytopic critical regions. Second, the memory footprint associated with the Chebyshev ball-based solution was evaluated for the same generated problems. In this thesis, the optimization problem formulation stated in (2.1) was considered.

First, 5 sets of large-scale systems were generated. Analogous to the case study presented in [32], each set represented a specific problem size and contained 5 different random systems. The differences in the sizes of the system sets were based on different numbers of the optimization variables, i.e., sizes of the control input vectors u. Every set had the same prediction horizon length $N = 45$ and dimension of the parameter $n = 50$. The summary of the problem sizes for every set of mpQP problems is summarized in Table 2.1.

Particularly, the memory footprint was determined for each of the 25 large-scale problems using both methods. For each of the investigated problems, the sets of

Table 2.1: Problem sizes of the generated sets of large-scale systems.

Set of systems	Dimension of u	Optimization variables	Constraints
1	17	765	6 030
2	20	900	6 300
3	22	990	6 480
4	25	1 125	6 750
5	30	1 350	7 200

random feasible seeding points were generated to obtain 300 unique critical regions, i.e., $\mathcal{R}_i$, $i = 1, \ldots, 300$. Note, to obtain 300 unique critical regions it is necessary to generate more than 300 initial seeding points as multiple initial conditions may lead to the same critical region, see Remark 2.3.2. Then, the average values of the memory footprints were analyzed for each set of 5 large-scale systems using (2.29) and (2.31).

We recall that in both cases, it is necessary to store the matrices of the optimization problem H, G, S, w, the binary vectors defining the active constraints $\mathcal{I}$ and the centers C of Chebyshev balls inscribed in every critical region of the partial solution. The difference lies in storing the Chebyshev balls' radii r when considering the second approach. The second difference in memory footprint is storing the critical regions in the form of polytopes in the original approach, see Theorem 2.3.4 and Theorem 2.4.4.

The generated results for the considered set of 5 types of large-scale systems are summarized in Table 2.2. The results presented in Table 1 were generated using MATLAB R2020b, YALMIP R20200930 [47] and solver GUROBI 9.1. The results were performed on a computer running 8 cores and AMD Ryzen 7 PRO 4750U at 4.1 GHz, and 16 GB RAM.

The first two columns of Table 2.2 provide the information about the given problem size in (2.12). The second column contains the information about the memory footprint of the optimization problem matrices H, G, S, w, which is the same for both approaches. The next two columns compare the memory footprint necessary to store the solution considering the polytopic approach (using Lemma 2.3.3) and the novel fixed-memory approach (using Lemma 2.4.2). Finally, the last two columns focus on memory savings which arise from the fixed-memory approach, while the first column contains the memory savings of the solution (using Lemma 2.4.2), and the second one contains the total memory savings considering also the problem matrices (using Theorem 2.4.4).

The generated results collected in Table 2.2 confirmed that considering the proposed fixed memory approach significantly outperformed the original method presented in [32].

Table 2.2: Results generated for the partial solution of 300 critical regions: memory footprint comparison using the polytopic regions and Chebyshev balls for different problem sizes.

Set of systems	Problem matrices [kB]	Solution memory footprint [kB]		Memory savings [%]	
		Polytopes	Chebyshev balls	Solution	Total
1	44 578	31 813	349	98.9	41.2
2	55 007	33 056	359	98.9	37.3
3	62 445	32 781	365	98.9	34.2
4	74 332	39 472	376	99.0	34.6
5	96 088	35 312	392	98.9	26.7

When considering just the solution of the partial explicit MPC, the memory footprint necessary to store the fixed memory solution composed of the Chebyshev balls centers C_i and radii r_i, and corresponding binary vectors indicating active constraints $\mathcal{I}_i$ is dramatically reduced compared to the memory footprint needed to store the polytopic solution composed of the polytopes $\mathcal{R}_i$, binary vectors $\mathcal{I}_i$, and centers of the Chebyshev balls C_i, $\forall i = 1, \ldots, 300$.

The memory footprint necessary to store the solution considering the fixed memory approach requires only around 1 % of the memory footprint corresponding to the original method, i.e., the memory savings of fixed memory partial solution reach up to 99 %, see Table 2.2.

Note that also the matrices H, G, S, w of large-scale optimization problem in (2.12) have to be stored to evaluate the online phase, see inputs of Algorithms 2, 4. The contribution of the large-scale optimization problem matrices H, G, S, w to the total memory consumption is significant and given, see Theorems 2.3.4, 2.4.4. Therefore, it is necessary to evaluate the memory savings also considering the large-scale optimization problem matrices H, G, S, w.

With increasing problem size, the contribution of the large-scale optimization problem matrices to the total memory footprint significantly increases, see Table 2.2, column "Problem matrices". Therefore, the gap between the total memory footprints of the considered two methods is lower, see Table 2.2, column "Total memory savings". Nevertheless, considering the smallest problem size (765 optimization variables and 6 030 constraints), when the problem data requires the lowest memory, the memory savings are the highest and reach up to 41.2 %, see Table 2.2, column "Total memory savings".

To conclude, the generated results in Table 2.2 demonstrate that the proposed fixed-

memory partial explicit MPC is not only an effective approach to determine the size of the partial solution in advance, but this method also provides a significant improvement in the memory savings compared to the original method in [32].

Chapter 3

Self-tunable model predictive control

The third chapter of this thesis is dedicated to the topic of self-tunable approximated explicit MPC. This control strategy addresses the second group of complex systems considered in this thesis – the nonlinear and asymmetric systems. In the first part, the concept of the tunable approximated explicit MPC is summarized. In order to be able to practically implement a tunable MPC, the self-tuning technique itself is elaborated. Two strategies of the tuning parameter setup are introduced. The first idea is based on the current reference value, and the second approach exploits the current reference change. Moreover, further scaling of the tuning parameter is proposed. Finally, the next part is dedicated to the experimental case study on a laboratory heat exchanger.

3.1 Tunable explicit model predictive control

Explicit model predictive control [5] utilizes a parametric solution of the model predictive control introducing its application range towards the systems with fast dynamics. Moreover, the explicit solution enables providing rigorous analysis and certification of the closed-loop system stability, constraints satisfaction, etc. As the explicit solution is available, real-time solving of the optimization problem in every control step is omitted, see Section 2.2. As this thesis chapter deals with industrial-oriented implementation of

the linear MPC, let us consider the MPC optimization problem in the following form:

$$\min_{u_0,u_1,\ldots,u_{N-1}} \quad \sum_{k=0}^{N-1} \Big((y_k - y_{\mathrm{ref}})^\intercal Q_{\mathrm{y}} (y_k - y_{\mathrm{ref}}) + u_k^\intercal R u_k + x_{\mathrm{I},k}^\intercal Q_{\mathrm{I}} x_{\mathrm{I},k} \Big) \tag{3.1a}$$

$$\text{s.t.:} \quad \widetilde{x}_{k+1} = \widetilde{A}_{\mathrm{d}}\, \widetilde{x}_k + \widetilde{B}_{\mathrm{d}}\, u_k, \tag{3.1b}$$

$$y_k = \widetilde{C}_{\mathrm{d}}\, \widetilde{x}_k, \tag{3.1c}$$

$$u_k \in \mathbb{U}, \tag{3.1d}$$

$$y_k \in \mathbb{Y}, \tag{3.1e}$$

$$\widetilde{x}_0 = \theta, \tag{3.1f}$$

$$k = 0, 1, \ldots, N-1, \tag{3.1g}$$

where k denotes the step of the prediction horizon N. To obtain the offset-free control results, the built-in integrator was included in the state-space model, e.g., see [74]. The prediction model in (3.1b)–(3.1c) has the form of augmented linear time-invariant (LTI) system for a given augmented state matrix $\widetilde{A}_{\mathrm{d}} \in \mathbb{R}^{n_{\tilde{\mathrm{x}}} \times n_{\tilde{\mathrm{x}}}}$, augmented input matrix $\widetilde{B}_{\mathrm{d}} \in \mathbb{R}^{n_{\tilde{\mathrm{x}}} \times n_{\mathrm{u}}}$ and augmented output matrix $\widetilde{C}_{\mathrm{d}} \in \mathbb{R}^{n_{\mathrm{y}} \times n_{\tilde{\mathrm{x}}}}$. Variables $\widetilde{x} \in \mathbb{R}^{n_{\tilde{\mathrm{x}}}}$, $u \in \mathbb{R}^{n_{\mathrm{u}}}$, $y \in \mathbb{R}^{n_{\mathrm{y}}}$ are vectors of corresponding augmented system states, control inputs, and system outputs, respectively. The sets $\mathbb{U} \subseteq \mathbb{R}^{n_{\mathrm{u}}}$, $\mathbb{Y} \subseteq \mathbb{R}^{n_{\mathrm{y}}}$ are convex polytopic sets of physical constraints on inputs and outputs, respectively. These sets include the origin in their strict interiors. The penalty matrix $Q_{\mathrm{y}} \in \mathbb{R}^{n_{\mathrm{y}} \times n_{\mathrm{y}}}, Q_{\mathrm{y}} \succeq 0$ penalizes the squared control error, i.e., the deviation between the controlled output and output reference value y_{ref}. The matrix $R \in \mathbb{R}^{n_{\mathrm{u}} \times n_{\mathrm{u}}}, R \succ 0$ penalizes the squared value of control inputs. The value of integrator is also penalized in the cost function with the penalty matrix $Q_{\mathrm{I}} \in \mathbb{R}^{n_{\mathrm{y}} \times n_{\mathrm{y}}}, Q_{\mathrm{I}} \succeq 0$. All the penalty matrices are considered to be diagonal due to the applicability of the self-tunable explicit MPC approach. The parameter $\theta \in \Theta$ in (3.1f) represents the initial condition of the optimization problem for which it is parametrically pre-computed.

The augmented model of the controlled system with the built-in integrator in (3.1b)–(3.1c) is rewritten as follows:

$$\widetilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ x_{\mathrm{I},k+1} \end{bmatrix} = \begin{bmatrix} A_{\mathrm{d}} & 0 \\ -T_{\mathrm{s}} C_{\mathrm{d}} & I \end{bmatrix} \begin{bmatrix} x_k \\ x_{\mathrm{I},k} \end{bmatrix} + \begin{bmatrix} B_{\mathrm{d}} \\ I \end{bmatrix} u_k, \tag{3.2a}$$

$$y_k = \begin{bmatrix} C_{\mathrm{d}} & 0 \end{bmatrix} \begin{bmatrix} x_k \\ x_{\mathrm{I},k} \end{bmatrix}, \tag{3.2b}$$

where $x_{\mathrm{I}} \in \mathbb{R}^{n_{\mathrm{y}}}$ is the integral of the control error, T_{s} denotes the sampling time, and

matrices A_d, B_d, C_d are the well-known state-space matrices that form the augmented LTI model. As a consequence of this extension and penalization in the cost function in (3.1a), not only the control error is penalized, but also the integrated value, which leads to analogous offset-free reference tracking results as incorporating an integral part in the PID controller.

The parametric solution of the optimization problem of the quadratic programming (QP) in (3.1) leads to the explicit solution in the form of piecewise affine PWA control law defined above the domain consisting of r critical regions:

$$u(\theta) = \begin{cases} F_1\,\theta + g_1 & \text{if} \quad \theta \in \mathcal{R}_1, \\ F_2\,\theta + g_2 & \text{else if} \quad \theta \in \mathcal{R}_2, \\ \qquad\vdots & \\ F_r\,\theta + g_r & \text{else if} \quad \theta \in \mathcal{R}_r, \end{cases} \tag{3.3}$$

where $F_i \in \mathbb{R}^{n_\mathrm{u} \times n_\mathrm{x}}$ and $g_i \in \mathbb{R}^{n_\mathrm{u}}$ respectively are the slope and affine section of the corresponding control law. The PWA function defined in (3.3) is stored and recalled in the online phase, i.e., during the real-time control. Based on the identified polytopic critical region $\mathcal{R}_i$, where the parameter θ belongs, the optimal control input is calculated according to the associated control law in (3.3).

Many other formulations of the optimization problems for the explicit MPC design were formulated mainly in terms of the definition of the cost functions in (3.1a). Also, the incremental (velocity) formulation of the state-space model is common, but leads to further extension of the vector of parameters θ, and therefore also the complexity of the explicit MPC controller increases. Another option for offset-free tracking is introducing disturbance modeling and estimation. For such an overview see, e.g., [35].

The aggressivity of the controller and the whole nature of the control is influenced by appropriate fine-tuning of the penalty matrices in the optimization problem in (3.1). When the multi-parametric QP (mpQP) problem is precomputed offline to obtain the corresponding parametric solution, it is not possible to tune the controller afterward without trading off a significant increase in the controller complexity or the performance loss. As the operating conditions and requirements on controller setup may differ throughout the control, the ability to adjust the controller's aggressivity can be very beneficial.

The idea of approximated tunable explicit MPC comes from the work [36], where the control action is calculated based on linear interpolation between two boundary

control actions. These control actions result from evaluating two boundary explicit MPCs. The boundary explicit controllers are constructed by solving the optimization problem having the same structure and setup, except for one of the penalty matrices – the tuned one. Based on the specific control application, any penalty matrix can be chosen as the tuned parameter, i.e., this approach is applicable for any penalty matrix. The boundary penalty matrices follow the assumptions on the penalty matrices from Section 2.1 and are diagonal matrices such that $\lambda_{i,\mathrm{L}} \leq \lambda_{i,\mathrm{U}}$, $\forall i = 1, \ldots, s$, where λ denotes the vector of eigenvalues of the penalty matrix, s is the rank of the tuned penalty matrix, and the subscripts L, U denote the lower and upper boundary setup, respectively.

Let us consider the penalty matrices in the cost function in (3.1a). Then, the penalty matrices are scaled in the following way:

$$R(k) = (1 - \rho(k))\, R_{\mathrm{L}} + \rho(k)\, R_{\mathrm{U}}, \tag{3.4a}$$
$$Q_{\mathrm{I}}(k) = (1 - \rho(k))\, Q_{\mathrm{I,L}} + \rho(k)\, Q_{\mathrm{I,U}}, \tag{3.4b}$$
$$Q_{\mathrm{y}}(k) = (1 - \rho(k))\, Q_{\mathrm{y,L}} + \rho(k)\, Q_{\mathrm{y,U}}, \tag{3.4c}$$

where ρ represents the tuning parameter such that $0 \leq \rho \leq 1$ holds. Based on the tuning rules in (3.4), it is possible to choose online any controller setup from the lower to the upper boundary of the tuned matrix. From the implementation point of view, it is preferred to tune just a single penalty matrix, i.e., to store only two controllers corresponding to the boundary values of the selected penalty matrix. To determine which penalty matrix in (3.4) should be tuned, it is suggested to judge the control performance by systematic tuning of all the penalty matrices. Systematic tuning involves selecting a specific penalty matrix and observing the control results by gradually increasing or decreasing the diagonal elements of the matrix. This procedure is then repeated for the remaining penalty matrices in a similar manner.

When the tuning parameter ρ is determined based on the current control conditions, the approximated optimal control action is evaluated using the two optimal controllers. Based on the boundary control actions, the interpolated, i.e., tuned control action is calculated using the convex combination:

$$u(k) = (1 - \rho(k))\, u_{\mathrm{L}}(k) + \rho(k)\, u_{\mathrm{U}}(k), \tag{3.5}$$

where u_{L} and u_{U} denote the optimal control actions from the lower and upper boundary explicit MPC, respectively. The online tuning comes with the cost of storing and

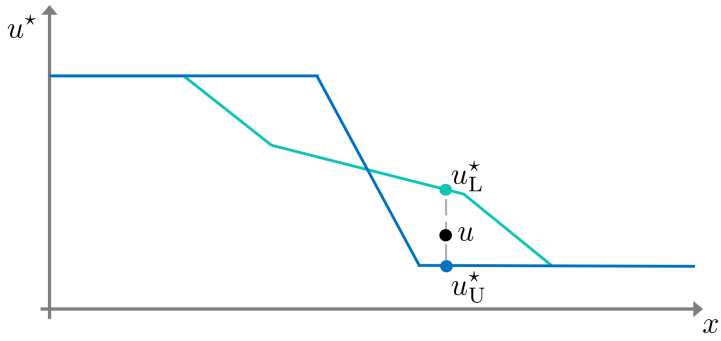


Figure 3.1: Principle of tunable explicit MPC. The final control input value is interpolated based on the upper (blue) and lower (green) boundary controller.

evaluating two explicit controllers. Nevertheless, the ability to tune the controller may be more important in many practical applications. The principle of tunable explicit MPC is depicted in Figure 3.1.

The concept of explicit MPC tuning is applicable to a wide class of MPC design formulations, based on the current specific control requirements. Without loss of generality, hereafter, let us consider the penalty matrices of the cost function in (3.1a), as it is necessary to satisfy offset-free reference tracking.

Remark 3.1.1. *If the asymptotic stability and recursive feasibility guarantees are required, the reader is referred to follow the instructions from [63]. In order to satisfy these requirements, the study introduces a procedure for computing the standard terminal penalty and terminal set for the two boundary controllers.*

Remark 3.1.2. *Not only* (3.5) *must be chosen for the interpolation of the final control input. Another way to tune the control input is by using some nonlinear relation for the interpolation.*

3.2 Self-tunable explicit model predictive control

The advantage of a tunable controller brings a question of how to design the logic of setting the tuning parameter ρ. This section provides an overview of the techniques and ideas related to setting of the tuning parameter.

3.2.1 Tuning parameter based on the reference value

In this section, the idea of online self-tuning is summarized [19]. The concept of self-tuning provides the possibility to adjust the aggressiveness of the controller without the necessity to intervene and tune the penalty matrices during control.

The need for real-time controller tuning often arises from tracking a time-varying piece-wise constant (PWC) reference. Our work [19] focuses on adjusting the penalty matrix when the reference value is changed. The further the reference value is from the steady state, the more aggressively the controller is tuned. The idea behind the suggested scaling lies in compensation for the nonlinear behavior of the system.

Consider a single-input and single-output (SISO) system or a multiple-inputs and multiple-outputs (MIMO) system with completely decoupled pairs of the control inputs and the system outputs. Then, the procedure of controller tuning is based on evaluating the different operating points between the current value of the reference and the system steady-state value. This deviation is considered to scale the value of control action. First, the maximal admissible absolute value of the reference is defined. Analogous to the reference trajectory preview concept of MPC design, this value can be determined based on the general knowledge of the expected future reference values. Another suggestion is to set the maximal deviation $d_{\max}$ based on the constraints on system outputs:

$$d_{\max} = \max(|y_{\min}|, y_{\max}), \tag{3.6}$$

where the symbol $|.|$, hereafter, denotes the element-wise absolute value, $y_{\min}$ and $y_{\max}$ are respectively lower and upper bound on the output variable in the deviation form, i.e., zero (origin) corresponds to the system steady-state value. Using the information about the maximal possible deviation $d_{\max}$, the tuning parameter ρ can be calculated as the ratio between the current reference value and the maximal deviation:

$$\rho(k) = \frac{|y_{\text{ref}}(k)|}{d_{\max}}. \tag{3.7}$$

Based on (3.7), the property $0 \leq \rho \leq 1$ holds, as $|y_{\text{ref}}| \leq d_{\max}$. As a consequence, the parameter ρ represents a way how to normalize the deviation from the steady-

state value and is exploited to scale the control action or, implicitly, to tune the aggressiveness of the controller.

Note that the reference value must be reachable from the operating range to ensure that $0 \leq \rho \leq 1$ holds. Otherwise, the interpolated control action would be the "extrapolation" leading to the loss of guarantees on the input or state constraints satisfaction, etc.

When considering tuning the control action based on (3.5), a higher value of tuning parameter ρ leads to approaching the upper boundary controller and vice versa. When tuning, e.g., the matrix Q_{y} penalizing the control error, a higher ratio ρ would lead to more aggressive control actions. When operating with the reference value close to the system steady-state value, the parameter ρ decreases and the control profiles become sluggish.

Remark 3.2.1. *In general, the parameter $d_{\max}$ is a vector, as it depends on the size of the system outputs. If $d_{\max}$ is scalar, the parameter ρ is scalar as well and can be directly utilized to scale the control action. If multiple outputs are controlled, it is suggested to calculate the tuning parameter based on the maximal ratio as follows:*

$$\rho(k) = \max\left(\frac{|y_{\mathrm{ref}}(k)|}{d_{\max}}\right). \tag{3.8}$$

Note that the relations in (3.7) and (3.8) operate with the absolute value of the reference. It is not taken into account whether the reference value changed upwards or downwards with respect to the system steady-state value placed in the origin, i.e., whether $\Delta_{\mathrm{ref}}(k) = y_{\mathrm{ref}}(k) - y_{\mathrm{ref}}(k-1) > 0$ or $\Delta_{\mathrm{ref}}(k) < 0$. As many plants have nonlinear behavior with an asymmetric nature (different behavior when the process variable is rising or decreasing), the positivity or negativity of the reference change could be considered in the controller self-tuning procedure to improve the control performance.

3.2.2 Tuning parameter based on the size of reference change

This section extends the ideas of self-tunable explicit MPC in order to improve control performance. First, a different way of tuning parameter calculation is introduced. Furthermore, an extended self-tunable technique is presented to scale the tuning parameter for industrial-oriented applications, when it is beneficial to exploit a specific range of the tuning parameter in different operating conditions.

Our approach of self-tunable explicit MPC introduced in [19] suggested tuning based on the current reference value distance from the steady state. The aim is to compensate for the nonlinear behavior of a system when using a simple linear prediction model. This thesis provides also another useful way of the real-time evaluation of the tuning parameter ρ based on the size of reference change. When different sizes of reference step changes are made and the behavior of the closed-loop system is varying, it can be beneficial to include the size of the reference step change in the tuning procedure.

In this approach, the aggressivity is adjusted based on the ratio between the reference step change and the maximal reference step change that can be realized during the control operation:

$$\rho(k) = \frac{|\Delta_{\mathrm{ref}}(k)|}{\Delta_{\max}}, \tag{3.9}$$

where $\Delta_{\mathrm{ref}}(k) = y_{\mathrm{ref}}(k) - y_{\mathrm{ref}}(k-1)$ is the size of the reference step change. The denominator of (3.9) is changed as well. In contrast to the maximal deviation from the steady state in Section 3.2, this approach introduces $\Delta_{\max}$ as the maximal possible reference step change. Analogously to the original approach, the maximal reference step can be set based on the general knowledge of the expected future reference values, i.e., $\Delta_{\max} = \|\Delta_{\mathrm{ref}}(k)\|_\infty, \forall k \geq 0$. Another option is to exploit the information about the system constraints and set the parameter $\Delta_{\max}$ according to (3.6).

Note, only the absolute value of Δ_{ref} and $\Delta_{\max}$ are considered in this procedure to ensure $\rho \geq 0$.

In (3.9), it is suggested to increase the value of tuning parameter ρ with increasing value of reference step change. Note, in this thesis, the larger value of the tuning parameter leads to adding more weight on the penalty matrices associated with the upper boundary controller, see (3.4). If the opposite logic of real-time controller tuning is requested, it is possible to adapt the tuning such that

$$R(k) = \rho(k)\, R_{\mathrm{L}} + (1 - \rho(k))\, R_{\mathrm{U}}, \tag{3.10}$$
$$Q_{\mathrm{I}}(k) = \rho(k)\, Q_{\mathrm{I,L}} + (1 - \rho(k))\, Q_{\mathrm{I,U}}, \tag{3.11}$$
$$Q_{\mathrm{y}}(k) = \rho(k)\, Q_{\mathrm{y,L}} + (1 - \rho(k))\, Q_{\mathrm{y,U}}, \tag{3.12}$$

hold. This change leads to adding more weight to the lower boundary controller with the increasing value of the tuning parameter ρ.

Remark 3.2.2. *The tuning parameter ρ should be updated only when the reference changes. Updating the tuning parameter in the control steps when $\Delta_{\mathrm{ref}} = 0$ would lead to using tuning parameter ρ with zero value, i.e., the control input would correspond to one boundary controller and would not be scaled.*

3.2.3 Self-tunable technique for systems with asymmetric behavior

This thesis provides a further extension of our self-tuning method proposed in [19]. The suggested technique of tuning is suitable, e.g., for systems with asymmetric behavior, but can be used in any application, where "simple" tuning in the whole range of tuning parameter ρ is not sufficient.

The proposed self-tuning method is based on splitting the interval of the tuning parameter ρ in order to utilize different parts of the interval in different operating conditions. Instead of the original value of tuning parameter ρ, the adjusted tuning parameter $\widetilde{\rho}$ is then utilized to scale the control input according to. (3.5).

Definition 3.2.1 (Decision function)**.** *For a given interval of tuning parameter ρ, $0 \leq \rho \leq 1$, let ρ_{s}, $0 < \rho_{\mathrm{s}} < 1$ be a boundary value splitting the interval into two parts. Let $\gamma : \mathbb{R} \to \mathbb{R}$ be an arbitrary function such that $0 \leq \gamma \leq 1$ holds. Then the decision function γ is constructed to assign its value either $\gamma \leq \rho_{\mathrm{s}}$ or $\gamma \geq \rho_{\mathrm{s}}$.*

Various decision functions γ can be considered. In this thesis, the decision functions according to (3.8) and (3.9) are suggested, while (3.9) was implemented and analyzed in the experimental case study in Section 3.3.

Definition 3.2.2 (Scaling of the tuning parameter)**.** *Given the value of tuning parameter ρ, $0 \leq \rho \leq 1$, the splitting value of the tuning parameter interval ρ_{s}, $0 < \rho_{\mathrm{s}} < 1$, and the value of the decision function γ, $0 \leq \gamma \leq 1$. Then the scaling of the tuning parameter $\widetilde{\rho}$ is given by:*

$$\widetilde{\rho} = \begin{cases} \rho\,\rho_{\mathrm{s}} & \text{if} \quad \gamma \in \langle 0, \rho_{\mathrm{s}} \rangle, \\ \rho\,(1 - \rho_{\mathrm{s}}) + \rho_{\mathrm{s}}, & \text{else if} \quad \gamma \in \langle \rho_{\mathrm{s}}, 1 \rangle. \end{cases} \tag{3.13}$$

Remark 3.2.3. *The introduction of splitting the tuning parameter $\widetilde{\rho}$ into the tuning intervals in* (3.13) *is not limited only to two intervals. If the nature of the controlled plant would benefit from splitting the operating range into more intervals, e.g., when the plant operates in the multiple steady-states values, then these intervals are simply*

determined by the corresponding values of $\rho_{\mathrm{s},i}$ for each part of the interval. Next, the tuning rules in (3.13) *are adopted in an analogous way.*

The following outcomes result from (3.13).

Lemma 3.2.4. *Given control law in* (3.3)*, its approximation given by the convex combination in* (3.5)*, and given scaled tuning parameter $\widetilde{\rho}$ according to Definition 3.2.2. Then the control action approximated into the form:*

$$u(k) = (1 - \widetilde{\rho}(k))\, u_{\mathrm{L}}(k) + \widetilde{\rho}(k)\, u_{\mathrm{U}}(k), \tag{3.14}$$

preserves the closed-loop system stability and recursive feasibility of the original control law in (3.3)*.*

Proof. It has been proven in [63] that for the asymptotic stable and recursive feasible pair of control inputs $(u_{\mathrm{L}}, u_{\mathrm{U}})$, the approximated control law in (3.3) preserves these properties for any ρ satisfying $0 \leq \rho \leq 1$, see Theorem 3.6 in [63]. It remains to prove that for any value of the scaled tuning parameter $\widetilde{\rho}$ according to the Definition 3.2.2 the same results hold. The rest of the proof of Lemma 3.2.4 consists of two parts corresponding to each particular rule in (3.13).
First, it is proved that the Lemma 3.2.4 holds for any $\gamma \leq \rho_{\mathrm{s}}$. Substituting a lower bound $\rho = 0$ into (3.13) leads to $\widetilde{\rho} = 0$. For the upper bound value of $\rho = 1$, from (3.13) holds $\widetilde{\rho} = \rho_{\mathrm{s}} < 1$. Next, for any value $0 < \rho < 1$ evaluation of the linear rule in (3.13) leads to the convex combination, i.e., $0 < \widetilde{\rho} < \rho_{\mathrm{s}}$ holds. Therefore, any value of $\widetilde{\rho}$ satisfies $0 \leq \widetilde{\rho} \leq \rho_{\mathrm{s}} < 1$. As a consequence, according to the Theorem 3.6 in [63], the asymptotic stability and recursive feasibility of the control law in (3.14) are preserved.
Secondly, it is proved that the Lemma 3.2.4 holds also for any $\gamma \geq \rho_{\mathrm{s}}$. Substituting a lower bound $\rho = 0$ into (3.13) leads to $\widetilde{\rho} = \rho_{\mathrm{s}}$. For the upper bound value of $\rho = 1$, from (3.13) holds $\widetilde{\rho} = 1$. Next, for any value $0 < \rho < 1$ evaluation of the linear rule in (3.13) leads to the convex combination, i.e., $\rho_{\mathrm{s}} < \widetilde{\rho} < 1$ holds. Therefore, any value of $\widetilde{\rho}$ satisfies $\rho_{\mathrm{s}} \leq \widetilde{\rho} \leq 1$. As a consequence, according to the Theorem 3.6 in [63], the asymptotic stability and recursive feasibility of the control law in (3.14) are preserved. □

Remark 3.2.5. *The Lemma 3.2.4 can be extended subject to the multiple intervals in an analogous way following the Remark 3.2.3.*

The advantage of the proposed method remains in the self-tuning of the controller as in the approach from Section 3.2. Nevertheless, it is required to appropriately determine

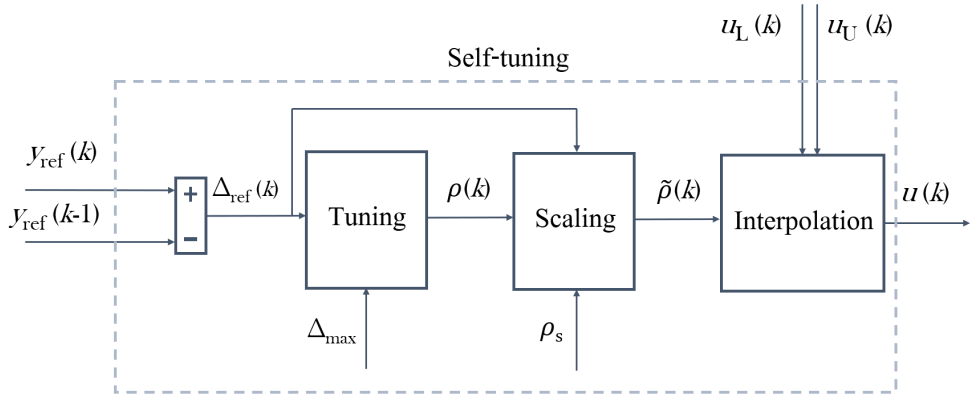


Figure 3.2: Scheme of the self-tuning control evaluation.

the splitting value of the tuning parameter ρ_{s} and assign the parts of the interval to the associated operating conditions. For a graphical overview of the proposed control technique, see the procedure of self-tuning evaluation depicted in Figure 3.2.

Remark 3.2.6. *Note, the suggested scaling method is suitable also for online MPC, as the optimization problem is solved in every control step. Therefore, it is possible to include the controller tuning in the procedure of computing the optimal control input.*

From the point of computational complexity, the proposed tuning procedure does not lead to any significantly demanding mathematical operations. Simple algebraic operations in (3.9) and (3.13) are evaluated online. Note, the overall control strategy still comes with the cost of storing and evaluating two explicit controllers.

3.3 Experimental results

In this section, the results of the proposed self-tuning method are analyzed by an laboratory implementation. The self-tuning strategy utilizes tuning parameter calculation based on the size of reference change (Section 3.2.2) and the scaling of tuning parameter based on splitting the interval of the parameter and assigning the interval parts to specific operating conditions (Section 3.2.3).

3.3.1 Heat exchanger plant

The plant on which the control was implemented and analyzed is a laboratory-scaled counter-current liquid-to-liquid plate heat exchanger Armfield Process Plant Trainer PCT23 [1], see Figure 3.3. The schematic of the plant is depicted in Figure 3.4. The heat exchanger is 90 mm wide, 103 mm long, and 160 mm high. The heat exchange is performed between the cold medium (water) and the hot medium (water). The cold medium as well as the heating medium are transported to the heat exchanger by two peristaltic pumps with flexible tubing from silicon rubber. The flow rate of the cold medium is constant, while the aim of control is to track the reference value of the outlet cold medium temperature. Therefore, the controlled variable is the cold medium temperature T at the outlet of the heat exchanger. The inlet cold medium temperature was constant during the whole control, i.e., $T_{\mathrm{C}} = 19^{\circ}\,\mathrm{C}$. The temperature of the heated cold medium in the outlet stream was measured by the type K thermocouple. The associated manipulated variable is the voltage U corresponding to the power of the pump feeding the heat exchanger by the hot medium. The voltage is within the range of $[0-5]\,\mathrm{V}$ normalized into the relative values in percentage. The maximal voltage $5\,\mathrm{V}$ or $100\,\%$ corresponds to volumetric flow rate $11.5\,\mathrm{ml\,s^{-1}}$. For further technical specifications of the laboratory heat exchanger, the reader is referred to [1]. As heat exchange is a nonlinear and asymmetric process [46], this heat exchanger represents a suitable candidate for the presented controller tuning strategy. The corresponding illustrative scheme of the implemented closed-loop control setup is in Figure 3.7, where the "Self-tuning" block substitutes the more detailed scheme of the tuning procedure in Figure 3.2.

The system model was identified by experimental identification. The aim was to work with linear nominal model in MPC optimization problem to decrease the numerical complexity. To avoid plant-model mismatch in order to ensure offset-free tracking, either disturbance observer or built-in integrator in (3.1) can be employed. Due to the ease of implementation, in this work, the built-in integrator was considered. The system was identified based on several measured step responses. The step changes were performed in the whole range of admissible values of manipulated variable and

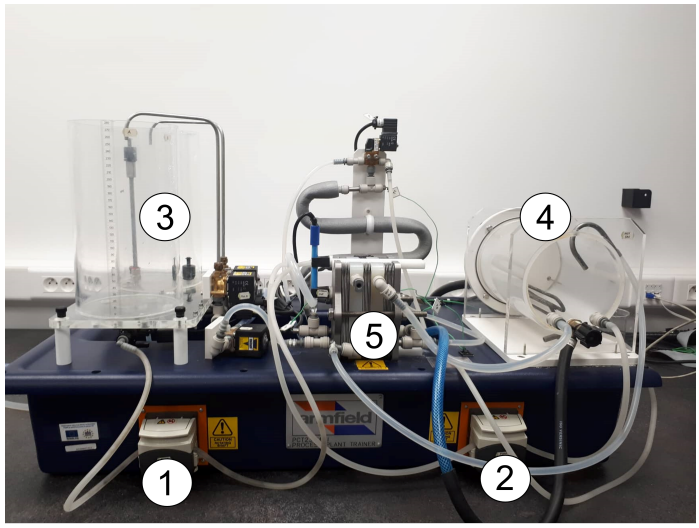


Figure 3.3: Laboratory heat exchanger Armfield Process Plant Trainer PCT23: cold medium pump (1), heating medium pump (2), cold medium tanks (3), heater for heating medium (4), heat exchanger (5).

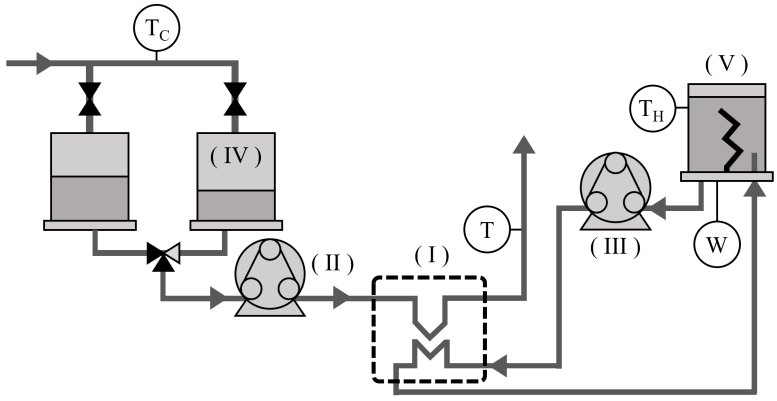


Figure 3.4: Scheme of Armfield PCT23. Heat exchanger (I), peristaltic pump for cold medium (II), peristaltic pump for heating medium (III), tank for cold medium (IV), heater for heating medium (V), temperature sensors (T – controlled temperature, T_C – cold outlet cold medium temperature, T_H – heating medium temperature), and electric power for maintaining the temperature of the heating medium (W).

every step response was identified by transfer function. It was possible to identify every step response as a first-order system, while the nominal gain and time constant are respectively $K = 0.24\,^{\circ}\mathrm{C}$ and $\tau = 5.7\,\mathrm{s}$. Finally, the nominal transfer function was converted to the state-space model. The matrices of the discrete-time state-space model of the plant are

$$A_{\mathrm{d}} = \begin{bmatrix} 0.839 \end{bmatrix}, \quad B_{\mathrm{d}} = \begin{bmatrix} 0.039 \end{bmatrix}, \quad C_{\mathrm{d}} = \begin{bmatrix} 1 \end{bmatrix}, \tag{3.15a}$$

considering the sampling time $T_{\mathrm{s}} = 1\,\mathrm{s}$.

3.3.2 Control design

This section delves into the control design. Besides the control design of two boundary explicit MPCs, it was necessary to keep the temperature of the heating medium constant. The heating medium was transported back to the heater after leaving the heat exchanger, i.e., the volume of the heating medium was recycled during the whole operation. The temperature of the heating medium was maintained on the value $70\,^{\circ}\mathrm{C}$ with a simple proportional controller with proportional gain $P = 20$. The control input from the proportional controller was the electric power which could acquire the values in the range $[0 - 2]\,\mathrm{kW}$ and was also normalized to percentage.

Next, the design of the boundary explicit model predictive controllers is elaborated. To respect the physical limitations of the operating conditions, the following constraints are considered in the terms of control inputs

$$-15\,\% \leq u \leq 65\,\%, \tag{3.16}$$

where the variable u represents the control inputs in the deviation form. The values of the heated cold medium temperature and voltage of the heating medium pump corresponding to zero steady states are respectively $T^{\mathrm{s}} = 35\,^{\circ}\mathrm{C}$ and $U^{\mathrm{s}} = 35\,\%$. Therefore, the physical constraints on the manipulated variable are actually

$$20\,\% \leq U \leq 100\,\%. \tag{3.17}$$

As the controlled system is naturally stable even if the maximal or minimal value of the manipulated variable is constantly applied, the constraints on the controlled variable in (3.1e) could be omitted. On the other hand, unbounded states/outputs

lead to higher memory consumption, because covering the whole possible range of parameters requires more critical regions. Therefore, the "redundant" constraints on the system outputs were included in order to reduce the number of critical regions and the overall memory footprint of the explicit controllers. The output constraints were set as:

$$-15\,^{\circ}\mathrm{C} \leq y \leq 20\,^{\circ}\mathrm{C}. \tag{3.18}$$

The constraints in (3.18) are equal to physical temperature as follows:

$$20\,^{\circ}\mathrm{C} \leq y \leq 55\,^{\circ}\mathrm{C}, \tag{3.19}$$

which corresponds to the range of temperature values which are achievable in the considered laboratory conditions and setup.

The penalty matrices of the problem in (3.1) were systematically tuned, and the corresponding control setup was implemented on the laboratory heat exchanger for each setup of the considered explicit MPC controllers. First, the tuning procedure aimed to determine which penalty matrix is the most suitable for real-time tuning. The most relevant was the penalty matrix Q_{y} as the tuning is directly associated with a reference value, which takes place in the calculation of the tuning factor ρ. Moreover, the tuning of Q_{y} preserved a satisfactory control performance. Next, the boundary values of the tunable matrix Q_{y} were tuned until the following limit values were determined based on the measured closed-loop control data: $Q_{\mathrm{y,L}} = 100$ and $Q_{\mathrm{y,U}} = 1\,000$. The built-in integrator was penalized with the fixed penalty matrix $Q_{\mathrm{I}} = 1$ and the control input with the fixed penalty matrix $R = 10$. The prediction horizon N was set to 20 control steps. The explicit model predictive controllers were constructed in MATLAB R2020b using the Multi-Parametric Toolbox 3 [23].

The explicit MPC corresponding to the penalty matrix $Q_{\mathrm{y,U}}$ contains 1 680 critical regions, and the explicit MPC with the penalty matrix $Q_{\mathrm{y,L}}$ contains 409 critical regions. The corresponding polytopic partitions can be seen in Figure 3.5 for the upper boundary controller and Figure 3.6 for the lower boundary controller.

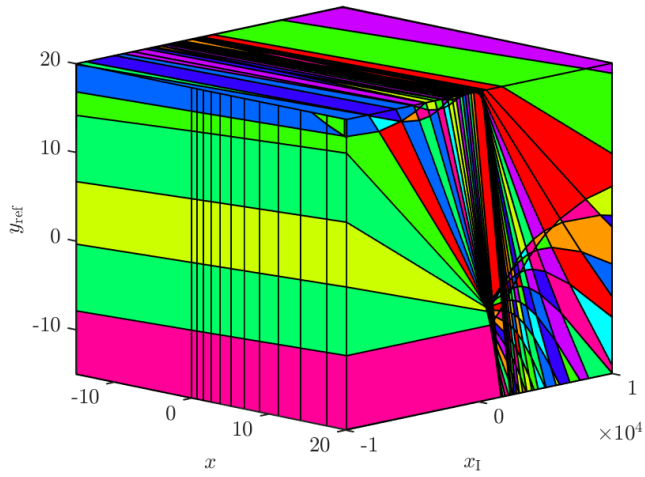


Figure 3.5: Polytopic partition of the upper boundary explicit MPC.

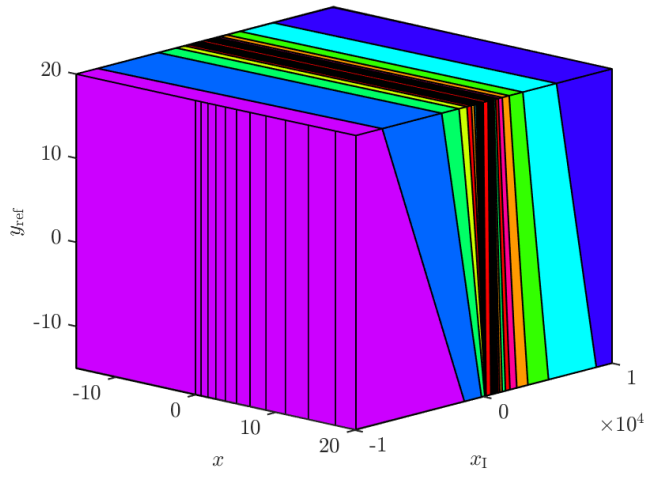


Figure 3.6: Polytopic partition of the lower boundary explicit MPC.

3.3.3 Control results

The designed explicit model predictive controllers were implemented to track a time-varying PWC reference. For the initial 200 seconds of control, the reference temperature was the steady-state value. After that, the reference changed its value twice upwards and twice downwards. The reference changes also acquired different sizes in order to examine the proposed tuning method as it is dependent on the size of the reference step change. Specifically, the reference temperature values were $T_{\text{ref}} = \{35\,^{\circ}\text{C}, 45\,^{\circ}\text{C}, 50\,^{\circ}\text{C}, 45\,^{\circ}\text{C}, 35\,^{\circ}\text{C}\}$.

The control profiles generated for both considered boundary control setups are compared in Figure 3.10 for the controlled variable, and in Figure 3.11 for the control inputs. Note, the constructed explicit MPC controller computed control inputs to respect the constraints on the control inputs and they need not be truncated afterward.

An interesting phenomenon can be observed while tracking the third reference value, i.e., $T_{\text{ref}} = 50\,^{\circ}\text{C}$. Although the steady-state values of temperature have the same value, the values of the manipulated variable are different. To confirm the correctness of the results, the measurements were performed multiple times and led to the same behavior. Also, the inlet temperatures of the cold and heating medium were checked to exclude the effect of a disturbance. Regarding the temperature of the cold medium, due to the limited hardware interface, it was not possible to measure the data continuously, store them, and plot the trajectory. Nevertheless, the temperature of the cold medium was manually measured multiple times during the experiment and was constant.

Regarding the temperature of the heating medium, the corresponding trajectories of the temperature can be seen in Figure 3.12, and the electric power, i.e., the corresponding manipulated variable, can be seen in Figure 3.13. Note that the legends correspond to the specific setup of MPC, but the temperature of the heating medium was controlled with a simple P controller with the same proportional gain in every control scenario. It can be seen in Figure 3.12 that the temperature of the heating medium remains relatively constant during the whole control, except for the undershoots in the scenario with upper boundary MPC, i.e., blue trajectory. The undershoots can be easily associated with the trajectory of the voltage on the pump dosing the heating medium (and ultimately the heating medium flow rate). As the upper boundary MPC calculated "aggressive" control inputs, the increased flow rate of the heating medium led to a slight decline in the heating medium temperature. After approximately 100 seconds, the heating medium warmed up to the reference value, i.e., $T_{\text{H,ref}} = 70\,^{\circ}\,\text{C}$ and remained constant within the accuracy of the temperature sensor. It can be seen that although the value of the voltage on the pump dosing the heating medium is different when tracking the temperature $T_{\text{ref}} = 50^{\circ}\,\text{C}$, the temperature of the heating medium is

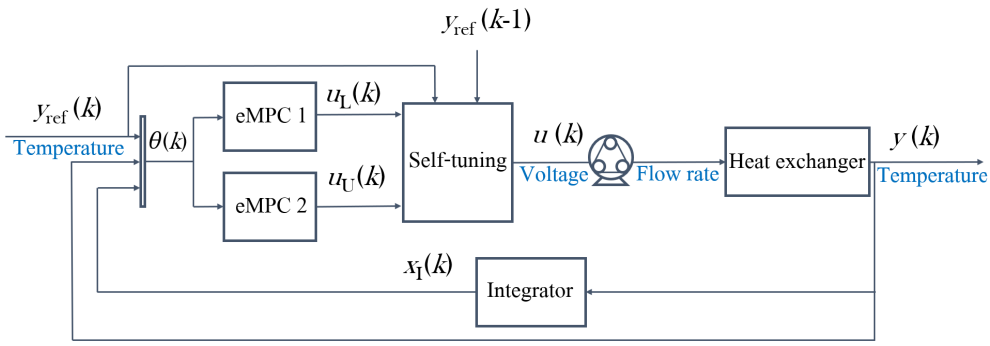


Figure 3.7: Scheme of the implemented closed-loop control setup, where "eMPC" denotes explicit MPC.

constant and identical for all control scenarios (MPC setups). Therefore, the same control conditions were fulfilled for all control scenarios.

The reason for this behavior could be explained by the peak of the manipulated variable associated with the upper boundary controller at time 800 s, see Figure 3.11, blue trajectory. After approximately 100 s, the value of the manipulated variable dropped and settled at a value lower than the value associated with the lower boundary controller, see Figure 3.11, red trajectory. This is a consequence of the heat accumulated inside the heat exchanger plates, and therefore, less heating medium was necessary to heat the cold medium. This phenomenon does not happen when tracking the reference value $T_{ref} = 45\,°C$, which originates in the nonlinear nature of the heat transfer process. When working in a higher temperature range, the gain of the heat transfer process decreases, and the sensitivity to changes in the heating medium flow is lower. Therefore, even different flow rates of the heating medium lead to the same temperature at the outlet.

The trajectories in Figure 3.10 show the asymmetric nature of controlling the plant of plate heat exchanger mainly when observing the overshoots and undershoots. When applying the control inputs associated with the lower boundary penalty matrix $Q_{y,L}$ in (3.1a), significant undershoots are present when tracking the reference downwards, i.e., when the reference change is negative. On the contrary, when implementing the controller associated with $Q_{y,U}$ in (3.1a), the undershoots are negligible, but significant overshoots can be seen when tracking the reference upwards, see Figure 3.10, blue trajectory.

These main experimental observations established the base for the strategy of controller self-tuning. The self-tuning strategy follows the ideas summarized in Section 3.2.2. Utilizing the nature of the boundary controller with the penalty matrix $Q_{\mathrm{y,L}}$ is preferred when the reference changes upwards. Therefore, in these operating conditions, the tuning factor is scaled in the first part of the whole interval, i.e., closer to the lower bound. On the contrary, tuning the controller closer to the upper boundary controller associated with $Q_{\mathrm{y,U}}$ is preferred for negative reference step changes. Therefore, in these operating conditions, the tuning factor is scaled above the splitting value ρ_{s}, i.e., closer to the upper bound. The splitting value of the tuning parameter was chosen simply in the middle of the interval, i.e., $\rho_{\mathrm{s}} = 0.5$. The remaining parameter that needed to be set was the maximal admissible size of the reference step change $\Delta_{\max}$, which was determined to 15 °C as the investigated range of controlled temperature was $[35 - 50]$ °C. Based on the aforementioned parameters and real-time information about the current reference change, the tuning factor was updated during control. The evolution of the scaled tuning factor $\widetilde{\rho}$ can be seen in Figure 3.8. When the positive reference changes are tracked, the tuning factor is scaled below the splitting value ρ_{s}. On the contrary, when the reference changes are negative, the tuning factor is scaled above the splitting value ρ_{s}.

The setup of the tuning factor can be associated with tuning of the penalty matrix Q_{y} according to (3.4c). The evolution of the penalty matrix Q_{y} during control is depicted in Figure 3.9. Note, the penalty matrix evolution in Figure 3.9 does not correspond to tuning of the optimal MPC, but serves for a deeper insight into the association of the interpolated control inputs with the optimal explicit MPC setup.

The control input is applied to the system each second, so there is a possible concern regarding the speed at which two explicit MPCs are evaluated. By analyzing the computational speed, it was concluded that the approximate control input can be generated in an average time of 0.01 seconds, which is 100-times faster than the sampling time.

The control results of the self-tunable technique compared with the boundary controllers can be seen in Figure 3.10 for the controlled variable, and in Figure 3.11 for the manipulated variable. It can be seen that the tuned controller combined the benefits of the two boundary controllers. The overshoots and undershoots were reduced, as in the first half of control the penalty matrix Q_{y} acquired value from the first half of the penalty interval. When tracking the reference with negative step change, the penalty matrix acquired the values from the second half of the interval, i.e., closer to the upper bound $Q_{\mathrm{y,U}}$. The similarity with the boundary controllers can be seen also on the manipulated variable profiles. Note, the constraints on the input variable

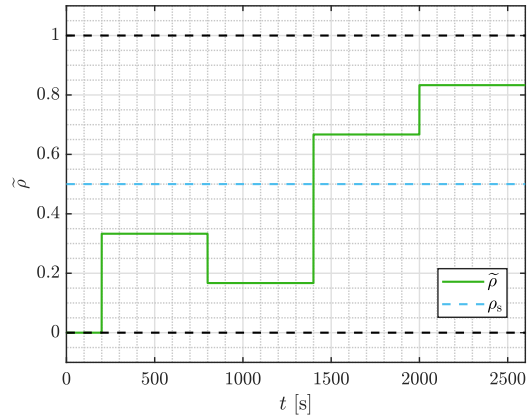


Figure 3.8: Evolution of the scaled tuning factor $\widetilde{\rho}$ during real-time control. When tracking positive reference changes, the tuning factor is scaled below the splitting value ρ_s (200 – 1 400 s). On the contrary, when the reference changes are negative, the tuning factor is scaled above the splitting value ρ_s (1 400 – 2 600 s).

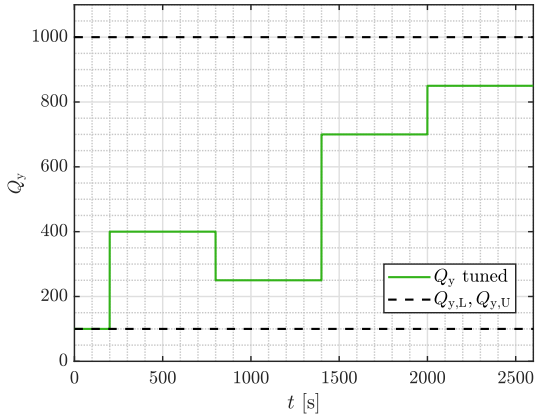


Figure 3.9: Evolution of the penalty matrix Q_y during real-time control. When tracking positive reference changes, the controller is tuned to operate closer to the lower boundary matrix $Q_{y,L}$ (200 – 1 400 s). On the contrary, when the reference changes are negative, the controller is tuned to operate closer to the lower boundary matrix $Q_{y,U}$ (1 400 – 2 600 s).

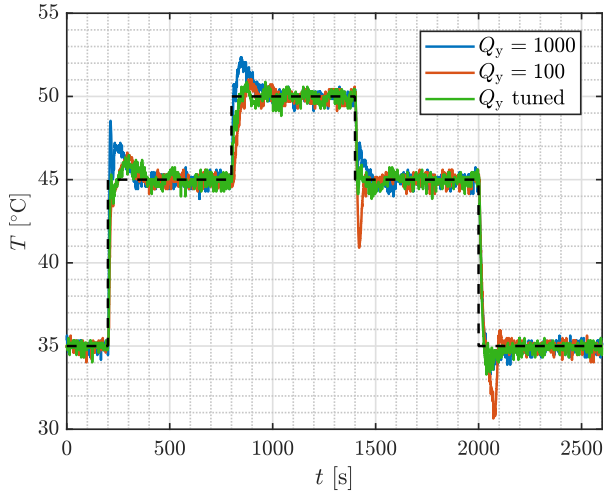


Figure 3.10: Explicit MPC: controlled variable trajectory for two boundary controllers and the tuned one. The solid lines represent the controlled temperature T and the dashed line represents the reference value.

were satisfied as they were scaled using linear interpolation based on the boundary controllers which are constructed considering the input constraints.

3.3.4 Control performance analysis

The control performance was also investigated quantitatively. Table 3.1 summarizes the evaluated control performance criteria computed for the two boundary controllers and the self-tuned controller. The control performance is evaluated for each reference step change separately. The considered quality criteria are: sum-of-squared control error SSE, maximal overshoot/undershoot $\sigma_{\max}$ and the settling time t_ϵ for 5%-neighbourhood of the reference temperature T_{ref}. To provide better readability of the computed results in Table 3.1, the best values, i.e., the minimum values, are emphasized using a bold font style.

As can be seen in Table 3.1, the real-time self-tuning of the explicit MPC controller helped to improve two to three criteria when tracking each reference value. The relative improvement in the percentage, denoted by δ, of using the self-tunable controller is summarized in Table 3.2 for each reference step change separately. The values were computed as the difference between two criteria values corresponding to the optimal and

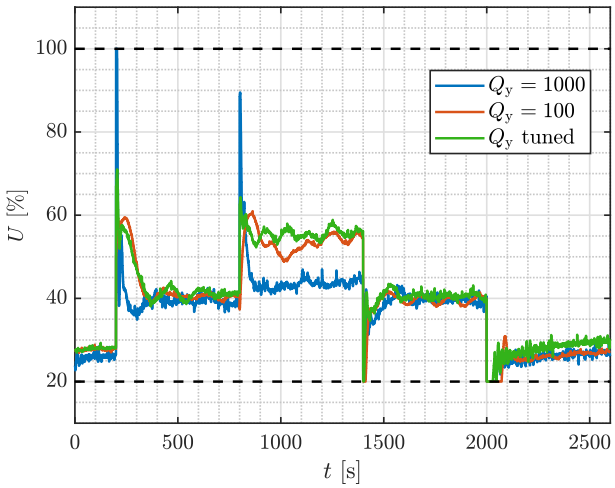


Figure 3.11: Explicit MPC: manipulated variable trajectory for two boundary controllers and the tuned one. The solid lines represent the voltage U and the dashed lines represent the constraints.

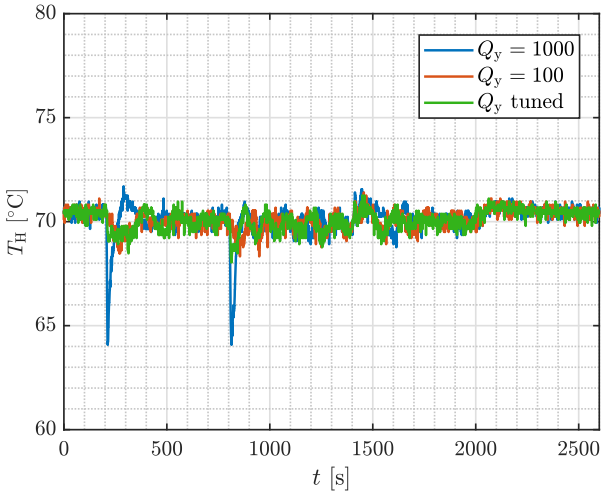


Figure 3.12: Auxiliary P controller: the trajectory of heating medium temperature control.

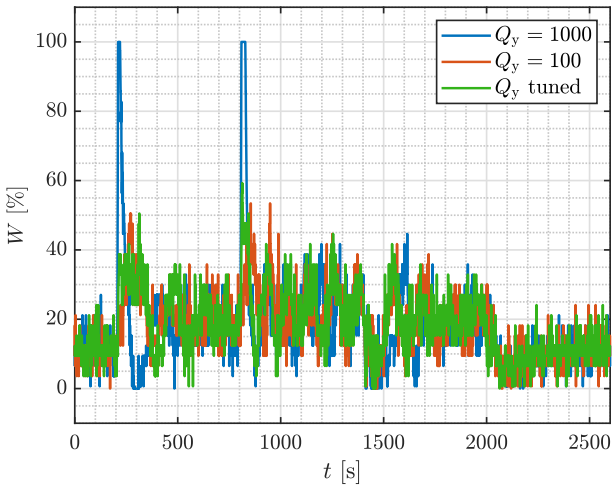


Figure 3.13: Auxiliary P controller: the trajectory of electric power controlling the heating medium temperature.

Table 3.1: Control performance criteria.

Reference step change	Q_y	SSE [°C^2 s]	σ_{max} [%]	t_ϵ [s]
35 °C → 45 °C	1 000	714	33.5	16.5
	100	867	16.7	12.5
	self-tuned	**678**	**15.2**	**9.5**
45 °C → 50 °C	1 000	365	47.2	**5**
	100	606	23.3	26.5
	self-tuned	**248**	**19.1**	9.5
50 °C → 45 °C	1 000	245	**18.9**	**6.5**
	100	398	79.6	31
	self-tuned	**186**	24.6	**6.5**
45 °C → 35 °C	1 000	1 024	18.4	22.5
	100	1 402	41.9	90
	self-tuned	**967**	**16.5**	**18.5**

self-tunable MPC, referred to the self-tunable MPC. The negative numbers represent deterioration of the specific performance criterion in the corresponding reference tracking.

Table 3.2: Relative improvement of the control performance using the self-tunable explicit MPC controller.

	Comparison with Q_y setup	δ SSE [%]	$\delta\sigma_{max}$ [%]	δt_ϵ [%]
35 °C → 45 °C	1000	5	121	74
	100	28	10	32
45 °C → 50 °C	1000	47	147	−47
	100	144	22	179
50 °C → 45 °C	1000	32	−23	0
	100	114	224	377
45 °C → 35 °C	1000	6	12	22
	100	45	154	386
Average	1000	23	64	12
	100	83	102	244

Compared to the considered non-self-tunable controllers, the control trajectories and the evaluated quality criteria confirmed the improved control performance for the reference tracking control problem of the heat exchanger with the non-linear and asymmetric behavior. Implementing a self-tunable explicit MPC controller leads to improved control performance in the most analyzed quality criteria, see Table 3.2. In average, the control performance criteria improved compared to the upper and lower boundary MPC respectively as follows: the squared-error-based criterion (SSE) reduced by 23 % and 83 %, the maximal overshoot/undershoot σ_{max} reduced by 64 % and 102 %, and the settling time t_ϵ reduced by 12 % and 244 %.

In general, utilizing the proposed controller with a scalable aggressiveness according to the operating conditions leads to higher accuracy (lower SSE), lower value of the overshoots (reduced σ_{max}), and faster achieving the reference value (decreased t_ϵ).

Obviously, if there exists a well-tuned "universal" controller that satisfies the requirements on the control performance in the whole range of the considered operating conditions, then the implementation of the self-tuning procedure is out of scope for such control application. Nevertheless, in numerous practical situations, using only one controller with a constant setup leads to poor or just "satisfactory" control results, i.e., the reference value is achieved, but with worse control performance, e.g., leading to high overshoots or settling times. When working on our laboratory case study, a set of different setups of penalty matrices was investigated. In every control scenario, the setup was beneficial only in some working conditions (tracking the reference upwards or downwards). Therefore, the closed-loop control performance is improved by introducing

the benefits of the self-tuning method based on the two boundary MPC controllers.

Note that this strategy relies on a proper design of the two boundary controllers. In case a non-negligible disturbance occurs, both boundary controllers should be able to solve a disturbance rejection problem as the final value of the manipulated variables is interpolated between them. To address the impact of the disturbances directly in constructing the MPC controller design, a robust MPC strategy should be considered. Any robust MPC design method leads to conservative control actions as some portion of the performance is sacrificed to compensate for the impact of the disturbances. Nevertheless, if it is possible to obtain the explicit (multi-parametric) solution of the robust explicit MPC offline, then the same self-tuning procedure is applicable to interpolate between the control actions from the robust controllers.

Chapter 4

MPT+ extension: implicit tube MPC

This chapter is devoted to the topic of implicit tube MPC, that is suitable for uncertain large-scale systems. When a large-scale system is considered, the tube MPC is not a suitable control technique, as it requires non-trivial set operations, challenging in higher state-space dimensions. A novel approach – implicit tube MPC – handles the MPC optimization problem differently, avoiding construction of sets, but still maintaining the optimal solution. The contribution of this thesis lies in incorporating of the implicit tube MPC into the MATLAB Multi-Parametric Toolbox extension, MPT+, to enable a wide usage, and user-friendly design and implementation of this control technique.
This chapter consists of the following parts: first, the theoretical background of tube MPC is recalled, and the concept of implicit tube MPC approach is explained. In the following part, the design procedure in MPT+ is introduced. Finally, the case studies validating the toolbox are presented.

4.1 Tube MPC

The tube MPC represents a robust control strategy which considers a bounded disturbance affecting the controlled system [56]. Given an uncertain, discrete-time, linear time-invariant (LTI) system defined as follows:

$$x(t+T_\mathrm{s}) = A_\mathrm{d}x(t) + B_\mathrm{d}u(t) + Ed(t), \tag{4.1}$$

t is the time sample of the discrete-time domain defined using a sampling time T_s. $A_\mathrm{d} \in \mathbb{R}^{n_\mathrm{x}\times n_\mathrm{x}}$ is system matrix, $B_\mathrm{d} \in \mathbb{R}^{n_\mathrm{x}\times n_\mathrm{u}}$ is input matrix, such that the matrix pair $(A_\mathrm{d}, B_\mathrm{d})$ is stabilizable. $E \in \mathbb{R}^{n_\mathrm{x}\times n_\mathrm{x}}$ is a disturbance matrix, $x \in \mathbb{R}^{n_\mathrm{x}}$ is the vector of the system states, $u \in \mathbb{R}^{n_\mathrm{u}}$ is the vector of the control inputs, $d \in \mathbb{R}^{n_\mathrm{x}}$ is a bounded

additive disturbance, for which the following holds:

$$w(t) = E\, d(t), \quad w(t) \in \mathbb{W}, \tag{4.2a}$$
$$\mathbb{W} = \left\{ w(t) \in \mathbb{R}^{n_\mathrm{x}} : \quad e_i^\top w(t) \leq 1, \quad i = 1, \ldots, N_\mathrm{w} \right\}, \tag{4.2b}$$

where N_w corresponds to the number of halfspaces representing the disturbance set.

Let us consider that the uncertain LTI system in (4.1) is constrained by

$$(x(t) \times u(t)) \in (\mathbb{X} \times \mathbb{U}), \tag{4.3}$$

where $\mathbb{X} \in \mathbb{R}^{n_\mathrm{x}}$ and $\mathbb{U} \in \mathbb{R}^{n_\mathrm{u}}$ are polytopes containing origin in their strict interiors.

Analogously to the disturbance set, the input and state constraints are reformulated to the normalized form:

$$\mathbb{S} = \mathbb{X} \times \mathbb{U} = \left\{ (x(t), u(t)) \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{u}} : \, c_l^\top x(t) + d_l^\top u(t) \leq 1, \, l = 1, \ldots, N_\mathrm{c} \right\}, \tag{4.4}$$

where N_c represents the number of polyhedral stage constraints on the inputs and states.

The conventional tube MPC optimization problem has the form [56]:

$$\min_{\hat{u}_0, \ldots, \hat{u}_{N-1}, \hat{x}_0, \ldots, \hat{x}_N} \quad \|\hat{x}_N\|_P^2 + \sum_{k=0}^{N-1} \left(\|\hat{x}_k\|_Q^2 + \|\hat{u}_k\|_R^2 \right) \tag{4.5a}$$
$$\text{s.t.:} \quad x(t) - \hat{x}_0 \in \mathbb{T}, \tag{4.5b}$$
$$\hat{x}_{k+1} = A_\mathrm{d} \hat{x}_k + B_\mathrm{d} \hat{u}_k, \tag{4.5c}$$
$$(\hat{x}_k \times \hat{u}_k) \in \left((\mathbb{X} \ominus \mathbb{T}) \times (\mathbb{U} \ominus K \cdot \mathbb{T}) \right), \tag{4.5d}$$
$$\hat{x}_N \in \mathbb{X}_\mathrm{N}, \tag{4.5e}$$

for $k = 0, \ldots, N-1$, and for prediction horizon N. Vectors $\hat{u}_k$ and $\hat{x}_k$ are decision variables optimized w.r.t. the nominal LTI system in (4.5c), without any perturbations of the disturbance w. The squared 2−norm objective function in (4.5a) is minimized for the symmetric positive definite penalty factors $Q \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$, $R \in \mathbb{R}^{n_\mathrm{u} \times n_\mathrm{u}}$, $P \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$. K represents the gain of a stabilizing controller, such that $A_\mathrm{d} + B_\mathrm{d} K$ is stable, e.g.,

LQR controller. The terminal set $\mathbb{X}_\mathrm{N}$ and the terminal penalty P are subject to the same assumptions as the terminal set $\mathbb{X}_\mathrm{N}$ and the terminal penalty P in optimization problem (2.4). The stability and recursive feasibility of the tube MPC optimization problem in (4.5) were proven in [56].

The convex set $\mathbb{T} \subset \mathbb{R}^{n_\mathrm{x}}$, denoted as a "tube", represents a way how to "robustify" the MPC, as only a disturbance-free prediction model is considered. It is constructed as a robust positively invariant approximation of the minimal robust positively invariant set following [68]:

$$\mathbb{T} = (1-\alpha)^{-1} \oplus_{j=0}^{N_\mathrm{S}-1} (A_\mathrm{d} + B_\mathrm{d}K)^j \, \mathbb{W}, \tag{4.6}$$

where symbol $\oplus$ denotes a Minkowski sum. The tube-*scaling* parameter α serves as a *safety* margin of the tube $\mathbb{T}$, what can be better seen in the following reformulation:

$$\mathbb{T} \ominus \alpha\mathbb{T} = \oplus_{j=0}^{N_\mathrm{S}-1} (A_\mathrm{d} + B_\mathrm{d}K)^j \, \mathbb{W}, \tag{4.7a}$$

$$\mathbb{T} = \oplus_{j=0}^{N_\mathrm{S}-1} (A_\mathrm{d} + B_\mathrm{d}K)^j \, \mathbb{W} \oplus \alpha\mathbb{T}. \tag{4.7b}$$

The tube $\mathbb{T}$ is iteratively constructed in N_S steps, such that the scalar α and finite integer N_S satisfy the following condition:

$$(A_\mathrm{d} + B_\mathrm{d}K)^{N_\mathrm{S}} \, \mathbb{W} \subseteq \alpha\mathbb{W}. \tag{4.8}$$

The parameter α is evaluated subject to a given tolerance $\alpha_\mathrm{tol} \in [0, 1), \alpha_\mathrm{tol} \ll 1$:

$$\alpha = \quad \max_i \quad \alpha_i \leq \alpha_\mathrm{tol}, \tag{4.9}$$

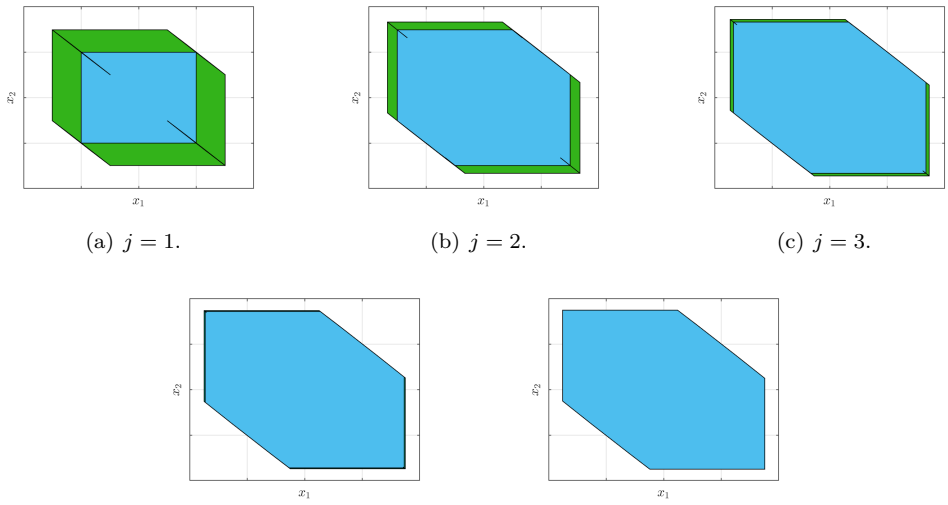
where

$$\alpha_i = \mathrm{h}\left(\mathbb{W}, \left((A_\mathrm{d} + B_\mathrm{d}K)^{N_\mathrm{S}}\right)^\top e_i\right), \quad i = 1, \ldots, N_\mathrm{w}. \tag{4.10}$$

In (4.10), the support function $\mathrm{h} : \mathbb{R}^n \times \mathbb{R}^{N_\mathrm{c}} \to \mathbb{R}^{N_\mathrm{c}}$ of a polyhedral set $\mathbb{W}$ for a general vector v is given by linear programming (LP):

$$\mathrm{h}(\mathbb{W}, v) = \max_w v^\top w, \tag{4.11a}$$

$$\text{s.t.} : e_i^\top w \leq 1, \quad i = 1, \ldots, N_\mathrm{w}. \tag{4.11b}$$



(a) $j = 1$. (b) $j = 2$. (c) $j = 3$.

(d) $j = 4$. (e) $j = 8$.

Figure 4.1: Example of construction procedure of the tube $\mathbb{T}$.

The tube construction procedure is demonstrated in Figure 4.1. The blue sets in Figures 4.1(a)–4.1(d) represent the starting sets in every iteration. The complements to the newly created sets are depicted in green. These complements are created by Minkowski addition of the term $(A_\mathrm{d} + B_\mathrm{d}K)^j\mathbb{W}$, see (4.6). The added set can be seen in every vertex of the starting blue sets. Although the set looks like a line segment, it is a full-dimensional set with a nonzero volume. Finally, Figure 4.1(e) represents the final tube after 8 steps, i.e., $N_\mathrm{s} = 8$.

The final control action $u(t)$ implemented to the uncertain LTI system in (4.1) is given by the control law $\kappa : \mathbb{X}_\mathrm{F} \to \mathbb{U}$

$$\kappa(x(t)) = \hat{u}_0^\star + K\left(x(t) - \hat{x}_0^\star\right), \tag{4.12}$$

where symbol $\star$ denotes the optimal solution of the MPC optimization problem in (4.5) and the feasibility set $\mathbb{X}_\mathrm{F} \subseteq \mathbb{R}^{n_\mathrm{x}}$ of the optimized initial condition $\hat{x}_0$ of (4.5) represents its domain. Further technical details can be found in [68, 67].

The actuators of the physical systems are often under limited rates on the control

actions formulated by:

$$\Delta u(t) = u(t) - u(t_-) \in \mathbb{U}_\Delta, \tag{4.13}$$

where $\Delta u(t) = u(t) - u(t_-)$ is the rate of control action and $\mathbb{U}_\Delta \in \mathbb{R}^{n_\mathrm{u}}$ is the corresponding convex set bounding the rates. Then, the tube MPC optimization problem in (4.5) is extended by the linear constraint having the form [25]:

$$\hat{u}_k + K(x(t) - \hat{x}_0) - u(t_-) \in \mathbb{U}_\Delta, \tag{4.14}$$

leading to the augmented vector of parameters $[x(t)^\top, u(t_-)^\top]^\top \in \mathbb{R}^{n_\mathrm{x}+n_\mathrm{u}}$, in case of the multi-parametric solution of (4.5).

4.2 Implicit tube MPC

The challenge of the tube MPC implementation lies in the repetitive set-based operations when constructing the tube (4.6), and in the constraints of the optimization problem handling this set, i.e., (4.5b) and (4.5d). The work [67] addresses this challenge to make the well-known tube MPC technique tractable for large-scale systems. The original set-based tube $\mathbb{T}$ is transformed into a corresponding implicit form $\mathcal{T}$ to avoid the set operations.

Analogous to the set-based tube $\mathbb{T}$ in (4.6), the implicit tube is given by

$$\mathcal{T} = (1-\alpha)^{-1} \sum_{j=0}^{N_\mathrm{S}-1} (A_\mathrm{d} + B_\mathrm{d} K)^j \, \omega_j, \tag{4.15}$$

where

$$e_i^\top \omega_j \leq 1, \quad i = 1, 2, \ldots, N_\mathrm{w}, \quad j = 1, 2, \ldots, N_\mathrm{S}. \tag{4.16}$$

Note, the uncertain parameters w in (4.2) are translated to the decision variables ω of the implicit tube MPC optimization problem to cover the iterative procedure of tube construction having N_S steps, i.e., $\omega_j \in \mathbb{R}^{n_x}$ for $j = 1, 2, \ldots, N_\mathrm{S}$. We recall, the implicit tube $\mathcal{T}$ is constructed without any set-based operations and has the form of

vector $\mathcal{T} \in \mathbb{R}^{n_x}$ determined by the time-invariant matrix pair A_d, B_d, controller gain K, and the sequence of uncertainties ω. This sequence corresponds to the iterative procedure of the tube construction, but in the implicit approach, the geometric tube is not constructed. Instead, the points ω of the original set $\mathbb{W}$ are considered, affected by N_s steps of the dynamics $(A_\mathrm{d} + B_\mathrm{d}K)$, which leads to the specific vertex of the original tube $\mathbb{T}$. Therefore, the initial condition changes from (4.5b) to the equality constraint:

$$x(t) - \hat{x}_0 = \mathcal{T}. \tag{4.17}$$

The difference between the original set-based tube $\mathbb{T}$ and the implicit tube vector $\mathcal{T}$ is depicted in Figure 4.2.

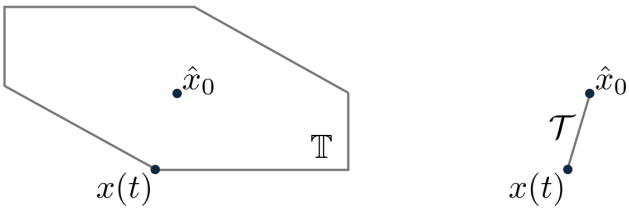


Figure 4.2: Comparison of the set-based tube $\mathbb{T}$ (left) and the implicit vector-based tube $\mathcal{T}$ (right).

Moreover, the implicit tube $\mathcal{T}$ is projected into the constraints in (4.5d) using a vector $f \in \mathbb{R}^{N_\mathrm{c}}$, such that the following two statements are equivalent:

$$(\hat{x}_k \times \hat{u}_k) \in ((\mathbb{X} \ominus \mathbb{T}) \times (\mathbb{U} \ominus K \cdot \mathbb{T}))\,, \tag{4.18a}$$
$$c_l^\top \hat{x}_k + d_l^\top \hat{u}_k \leq 1 - f_l, \quad l = 1, 2, \ldots, N_\mathrm{c}. \tag{4.18b}$$

The vector $f \in \mathbb{R}^{N_\mathrm{c}}$ represents the implicit tube projection into the state and input constraints *shrinking* the volume of the original sets $\mathbb{X}$ and $\mathbb{U}$. The vector f is constructed offline and given by:

$$f_l = (1-\alpha)^{-1} \sum_{j=0}^{N_\mathrm{S}-1} \mathrm{h}\left(\mathbb{W}, \left((A_\mathrm{d} + B_\mathrm{d}K)^j\right)^\top \eta_l\right), \tag{4.19a}$$
$$\eta_l = c_l + K^\top d_l, \quad l = 1, 2, \ldots, N_\mathrm{c}. \tag{4.19b}$$

Analogously, the terminal constraint in (4.5e) is reformulated using the vector f:

$$(c_l^\top + d_l^\top K)\hat{x}_\mathrm{N} \leq 1 - f_l, \quad l = 1, 2, \ldots, N_\mathrm{c}. \tag{4.20}$$

Note, the evaluation of the implicit tube $\mathcal{T}$ in (4.15) and of the vector f in (4.19) does not include the Minkowski sum operation, leading to high computational complexity or even impossibility to construct the sets in large-scale control applications. Therefore, by avoiding this operation, the implicit tube MPC form of (4.5) transformed according to [67] is given by a convex quadratic programming problem

$$\begin{aligned}
\min_{\substack{\hat{x}_0, \hat{u}_0, \omega_0, \ldots \\ \hat{u}_{N-1}, \hat{x}_N, \omega_{N_\mathrm{S}}}} \quad & \hat{x}_N^\top P \hat{x}_N + \sum_{k=0}^{N-1} \left(\hat{x}_k^\top Q \hat{x}_k + \hat{u}_k^\top R \hat{u}_k \right) && \text{(4.21a)} \\
\text{s.t.:} \quad & \hat{x}_{k+1} = A_\mathrm{d}\, \hat{x}_k + B_\mathrm{d}\, \hat{u}_k, && \text{(4.21b)} \\
& c_l^\top \hat{x}_k + d_l^\top \hat{u}_k \leq 1 - f_l, && \text{(4.21c)} \\
& \left(c_l^\top + d_l^\top K \right) \hat{x}_N \leq 1 - f_l, && \text{(4.21d)} \\
& x(t) - \hat{x}_0 = \mathcal{T}, && \text{(4.21e)} \\
& e_i^\top \omega_j \leq 1, && \text{(4.21f)} \\
& i = 1, 2, \ldots, N_\mathrm{w}, && \text{(4.21g)} \\
& j = 1, 2, \ldots, N_\mathrm{S}, && \text{(4.21h)} \\
& k = 0, 1, \ldots, N-1, && \text{(4.21i)} \\
& l = 1, 2, \ldots, N_\mathrm{c}. && \text{(4.21j)}
\end{aligned}$$

Finally, the control law remains the same as in the implicit tube MPC approach in (4.12).

Compared to the original tube MPC approach, the implicit tube MPC handles N_S more optimization variables caused by the additional sequence ω. Despite this fact, the implicit formulation does not limit the tube MPC only to geometrically simple problems.

4.3 Design procedure in MPT+

In this section, the procedure of tube MPC design in the MATLAB MPT+ toolbox [26] is described. The MPT+ toolbox is freely available on `GitHub`[1]. The package can be installed by setting the corresponding path in MATLAB or simply via `tbxManager`[2] by typing:

```
tbxmanager install mptplus
```

4.3.1 Tube MPC design

In this section, the well-known double integrator system is used to demonstrate the tube MPC design, as the two-dimensional state-space is suitable for graphical demonstration of the results. The discrete-time state-space respresenation of the double integrator system is defined as follows:

$$x(t+T_\mathrm{s}) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} d(t). \tag{4.22}$$

The constraints on the state and input variable, respectively, are defined as:

$$[-200, -2]^\top \preceq x(t) \preceq [200, 2]^\top, \tag{4.23a}$$
$$-1 \leq u(t) \leq 1, \tag{4.23b}$$

and the disturbance is constrained in a following way:

$$[-0.1, -0.1]^\top \preceq d(t) \preceq [0.1, 0.1]^\top. \tag{4.24}$$

The system in (4.22) together with the constraints in (4.23) and (4.24) is easily constructed as an object `model` as follows:

```
model = ULTISystem('A', [1, 1; 0, 1], 'B', [0.5; 1], 'E', [1, 0; 0, 1])
model.d.min = [-0.1; -0.1]
model.d.max = [ 0.1;  0.1]
```

[1] `MPTplus: https://github.com/holaza/mptplus`
[2] `tbxManager: https://www.tbxmanager.com`

```
model.x.min = [-200; -2]
model.x.max = [ 200;  2]
model.u.min = [-1]
model.u.max = [ 1]
```

Note, it is sufficient to set the constraint on every variable in a form of minimal and maximal admissible value, and it is not necessary to normalize the constraint set according to (4.4) and (4.2). Nevertheless, the possibility to set the normalized form of constraints is supported. The setup of the normalized constraint set is shown on the example of state constraints:

```
PX = Polyhedron('A',[1/200 0;0 1/2; -1/200 0; 0 -1/2],'b', ones(4,1))
model.x.with('setConstraint')
model.x.setConstraint = PX
```

If the constraints on the rate of change of the input variable are required, one can set them as follows:

```
model.u.with('deltaMin')
model.u.with('deltaMax')
model.u.deltaMin = -0.5
model.u.deltaMax =  0.5
```

Let us consider the prediction horizon $N = 9$ steps, and the following penalty matrices in (4.5):

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 0.1 \end{bmatrix}. \tag{4.25}$$

The penalty matrices Q and R and the prediction horizon N are created in MPT+ in a following way:

```
model.x.penalty = QuadFunction([1,0;0,1])
model.u.penalty = QuadFunction(0.01)
N = 9
```

Finally, the tube MPC controller is constructed:

```
option = {'solType',1,'LQRstability',1}
TMPC = TMPCController(model,N,option)
```

The first `option` parameter `solType` = 1 specifies the type of the returned optimized variables as the final control inputs $u(t)$ in (4.12), i.e., those which are sent directly to the controlled system in (4.1). Alternatively, if `solType` = 0, the output from the controller comes in a nominal form as a pair of $\hat{u}_0^\star$ and $\hat{x}_0^\star$. If the `solType` option is not specified, the default value is 1.

Secondly, the terminal set and terminal penalty are enforced through the options parameter `LQRstability` = 1. By setting the value to 1, the LQR-based terminal set and the Lyapunov penalty matrix are calculated and considered in the tube MPC. If the value of the parameter `LQRstability` is set to 0, then no terminal penalty and terminal set are automatically computed. The user is expected to set their own terminal penalty and terminal set. If the `LQRstability` option is not specified, the default value is 0.

The optimal control input for a given initial condition `x0` is evaluated as follows:

```
x0 = [ -5; -2 ];
u = TMPC.evaluate(x0)
```

In MPT+, similarly to the original MPT toolbox, it is possible to obtain data from the closed-loop simulation consisting of `Nsim` control steps.

```
Nsim = 12;
ClosedLoopData = TMPC.simulate(x0, Nsim)
```

It is also possible to specify the disturbance affecting the controlled system during the closed-loop simulation. If the disturbance sequence is not specified by user, it is randomly generated from the disturbance set.

```
ClosedLoopData = TMPC.simulate(x0, Nsim, w)
```

The `ClosedLoopData` contains useful data corresponding to the trajectories of control inputs, controlled states and the overall cost. If the `soltype` option is set to 0, then also trajectory of nominal states is available. The data can be accessed by typing:

```
ClosedLoopData.X
ClosedLoopData.U
ClosedLoopData.cost
ClosedLoopData.Xnominal % only for soltype=0
ClosedLoopData.Unominal % only for soltype=0
```

The trajectory of the controlled states of the double integrator system in state space is depicted in Figure 4.3. The time profiles of the controlled states and control inputs can be seen in Figure 4.4.

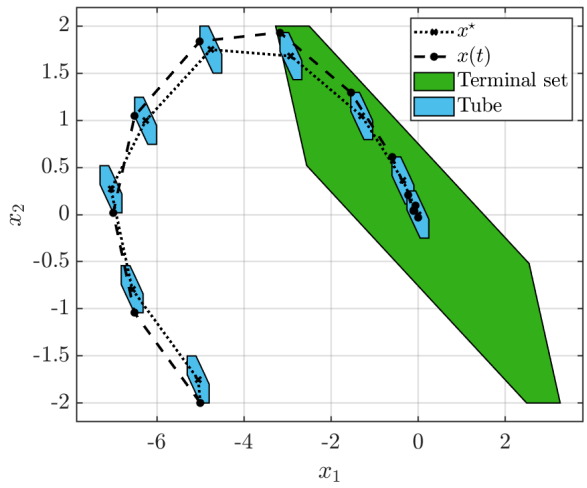


Figure 4.3: Closed-loop simulation of double itegrator control in the state space.

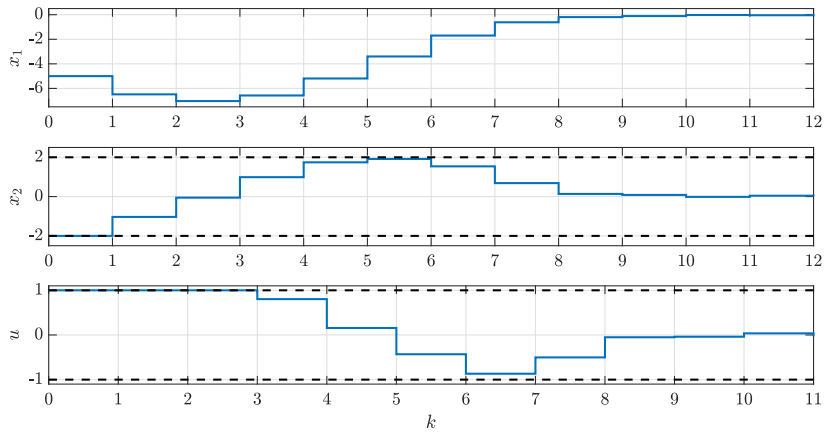


Figure 4.4: Closed-loop simulation of double itegrator control. The solid lines represent the corresponding variable x_1, x_2, u, and the dashed lines represent the constraints.

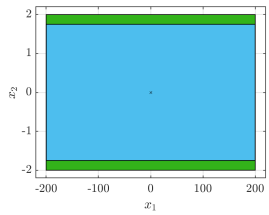
4.3.2 Handling the parameters and geometric sets

All the relevant parameters and sets related to the tube MPC can be easily handled. For example, the tube $\mathbb{T}$ is accessed and plotted via:

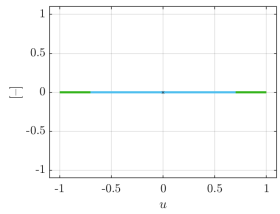
```
Tube = TMPC.TMPCparams.Tube
figure, Tube.plot()
```

Similarly, the sets corresponding to the state constraints, input constraints and disturbance set can be accessed. When handling the state and input constraints, one can choose between the original sets $\mathbb{X}$ and $\mathbb{U}$ or the conservative sets – the original sets after the subtraction of the tube, i.e., $\mathbb{X} \ominus \mathbb{T}$, $\mathbb{U} \ominus K \cdot \mathbb{T}$.

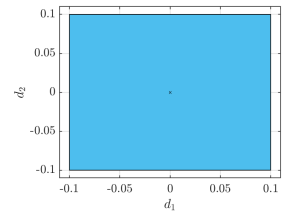
```
% State constraints
X = TMPC.TMPCparams.Px
Xconservative = TMPC.TMPCparams.Px_robust
% Input constraints
U = TMPC.TMPCparams.Pu
Uconservative = TMPC.TMPCparams.Pu_robust
```



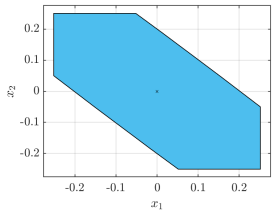
(a) Constraint set $\mathbb{X}$ (green) and $\mathbb{X} \ominus \mathbb{T}$ (blue)



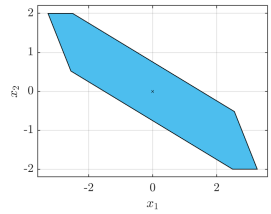
(b) Constraint set $\mathbb{U}$ (green) and $\mathbb{U} \ominus K \cdot \mathbb{T}$ (blue)



(c) Disturbance set $\mathbb{W}$



(d) Tube $\mathbb{T}$



(e) Terminal set $\mathbb{X}_\mathrm{N}$

Figure 4.5: The tube MPC design sets.

```
% Disturbances
Wset = TMPC.TMPCparams.Pw
```

It is possible to extract the stabilizing controller gain K and terminal penalty matrix P, which is useful mainly in the case when the parameters are not set manually, but the `LQRstability` option is set to 1. Moreover, the computed terminal set can be accessed as well:

```
% Controller gain
K = TMPC.TMPCparams.K
% Terminal penalty matrix
P = model.x.terminalPenalty.weight
% Terminal set
X_N = TMPC.model.x.terminalSet
```

The sets coresponding to the tube MPC design are depicted in Figure 4.5.

4.3.3 Implicit tube MPC design

In default, the tube MPC is constructed in the original – geometric manner, based on the Section 4.1. Otherwise, if the implicit tube MPC is required, it is necessary to set the `TubeType` option to `implicit` as follows:

```
option = {'TubeType','implicit','soltype',1,'LQRstability',1}
implicit_MPC = TMPCController(model,N,option)
```

The `TubeType` option supports also `explicit` property, which corresponds to the default setup, i.e., geometric sets construction and operations. Note, the explicit tube type does not correspond to explicit, i.e., multiparametric solution of the MPC. If the parametric solution of the tube MPC optimization problem is required, it is necessary to call the function `toExplicit` as in the original MPT toolbox:

```
eTMPC = TMPC.toExplicit
```

Although MPT+ supports the construction of the multiparametric controller from the implicit tube MPC using a function `toExplicit`, it does not make much sense to do so. The implicit tube MPC is beneficial in larger problem scales, when the geometric operations are not possible or impractical, which is a consequence of challenging size of the optimization problem. Therefore, it is also impractical to construct, if even possible, the multiparametric solution of the tube MPC. If the multiparametric solution of tube MPC is required, it is recommended to design the geometric-set-based tube MPC and transform it into the multiparametric form, when applicable.

Almost all of the above mentioned functionalities are possible also when considering the implicit tube MPC. Obviously, it is not possible to handle the tube and terminal set, which are not constructed. Instead, one can access the parameters α, α_{tol}, N_{s} and constraint vector f which are associated with the tube MPC design procedure. These commands are supported also for the original, set-based tube MPC.

```
% number of iterations in construction of the implicit tube: Ns
Ns = implicit_MPC.TMPCparams.Ns
% shrinking factor of the implicit tube: alpha
alpha = implicit_MPC.TMPCparams.alpha
% tolerance for evaluation of alpha: alpha_tol
alpha_tol = implicit_MPC.TMPCparams.alpha_tol
```

```
% vector robustifying the constraints: f
f = implicit_MPC.TMPCparams.f
```

4.4 Control implementation

Two case studies with different benchmark systems are presented in this section. The first case study contains experimental implementation of tube MPC on a simple SISO system Flexy^2 [31]. The aim of the first control implementation is to demonstrate the functionality of the toolbox in real-time control. The next case study involves a numerical simulation of implicit tube MPC implemented on a large-scale reactor-separator system adapted from [13]. The aim of the second case study is to demonstrate the validation of the implicit tube MPC implementation via MPT+ in a scenario when the original set-based tube MPC is not possible to construct.

4.4.1 Tube MPC implementation

To validate the set-based tube MPC incorporation to the MPT+ toolbox, an experimental case study was performed and published in [25]. The considered system was a SISO system with fast dynamics Flexy^2 [31], see Figure 4.6. The actuator is a fan that propels an air column in an upward vertical direction. The power of the airflow is measured by a flexible sensor placed in the air column. The sensor changes its electrical resistance according to the bend caused by the push of the air. Therefore, the flex sensor bend $b(t)$ in percentage is assumed as the controlled variable, and the fan speed $v(t)$ in percentage is a manipulated variable.

Flexy2 is a system with non-linear dynamics, as the sensitivity of the flex bend decreases when increasing fan speed. Moreover, the measurement noise is also present. These challenges make this device a suitable candidate for the implementation of tube MPC. As the system dynamics is naturally very fast and requires low sampling time with a fast evaluation of the control inputs, the explicit solution of tube MPC is considered. The goal is to control the flex sensor bend b to a steady-state value b^{s} and reject the effect of a disturbance.

The model of the system was obtained through experimental identification based on several step responses. The matrices of the nominal state-space system, transformed into the discrete-time domain using sampling time $T_{\mathrm{s}} = 0.01\,\mathrm{s}$ are:

$$A_{\mathrm{d}} = \begin{bmatrix} 0.966 \end{bmatrix}, \quad B_{\mathrm{d}} = \begin{bmatrix} 0.101 \end{bmatrix}. \tag{4.26}$$

As the controlled as well as the manipulated variable were set in percentage, their

Figure 4.6: Flexy^2 device [31].

values were constrained from the minimal value $b_{\min} = v_{\min} = 0\,\%$ up to the maximal value $b_{\max} = v_{\max} = 100\,\%$. Considering the steady-state values where the model was linearized, i.e., $v^{\mathrm{s}} = 40\,\%$ and $b^{\mathrm{s}} = 68\,\%$, the constraints were set in both presented control methods as follows:

$$-40\,\% \leq u(t) \leq 60\,\%, \quad -68\,\% \leq x(t) \leq 32\,\%. \tag{4.27}$$

Moreover, the change of the input variable was set to validate the control method implemented in the toolbox:

$$-55\,\% \leq \Delta u \leq 55\,\%. \tag{4.28}$$

As the uncertainties in the model were considered, the additive disturbance was defined as:

$$-1\,\% \leq d(t) \leq 1\,\%, \tag{4.29}$$

and the associated matrix E multiplying the vector of disturbances d in (4.1) was set as identity matrix.

By systematic tuning, the penalty matrices Q, R of the optimization problem in (4.5) were respectively assigned as:

$$Q = 10, \quad R = 1. \tag{4.30}$$

The prediction horizon N was set to 30 steps. Furthermore, the terminal set $\mathbb{X}_\mathrm{N}$ and the terminal penalty P were calculated by setting the option `LQRstability` $= 1$, and the evaluated values were:

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} x \leq \begin{bmatrix} 31.8694 \\ 21.2463 \end{bmatrix}, \quad P = 95.3852\,. \tag{4.31}$$

The explicit tube MPC was constructed in MATLAB 2020b programming environment, using toolboxes MPT 3.2.1, MPT+, YALMIP R20210331, and solver Gurobi 9.1.1. The tube explicit MPC was executed on CPU AMD Ryzen 7 PRO 4750U, 1.7 GHz with 16 GB RAM. From the viewpoint of computational complexity, the average time to evaluate the control input was 1 milisecond, which is suitable for the considered sampling time $T_\mathrm{s} = 0.01\,\mathrm{s}$.

After the construction of the tube explicit model predictive controller, the disturbance rejection control problem was investigated. The aim was to drive the flex sensor bend to the steady state $b^\mathrm{s} = 68\,\%$, while rejecting the effect of the two disturbances occurring at times 5 s and 10 s. The control results can be seen in Figure 4.7 for the controlled variable, i.e., the flex sensor bend $b(t)$. In Figure 4.8, the corresponding trajectory of manipulated variable is depicted, i.e., the trajectory of fan speed $v(t)$.

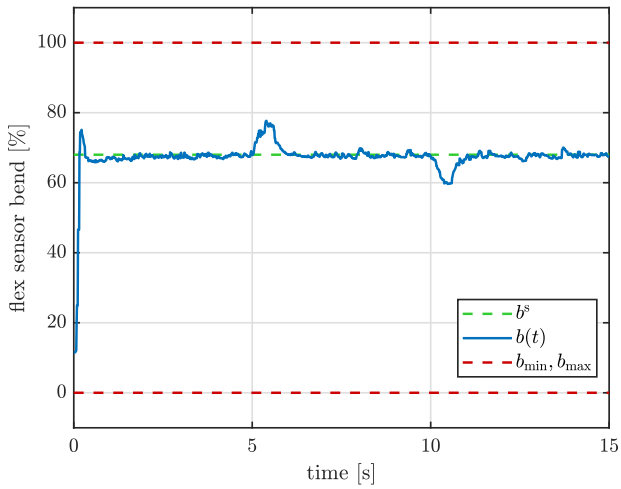


Figure 4.7: Tube model predictive control of Flexy^2 – controlled variable.

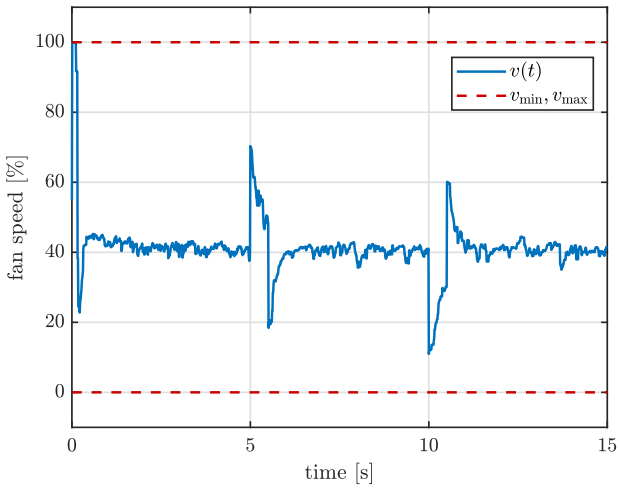


Figure 4.8: Tube model predictive control of Flexy^2 – manipulated variable.

When observing Figure 4.7 and Figure 4.8, it can be seen that the effects of the disturbances were rejected, with respect to the constraints on the manipulated and controlled variables. Moreover, the constraints on the change of the manipulated variable were active in the first control step and were satisfied as well. To conclude, the goal of the control was achieved, using the developed toolbox in real-time control.

4.4.2 Implicit tube MPC implementation

The suitablity of the implicit tube MPC is demonstrated on the large-scale reactor-separator system with 12 states and 3 inputs, adapted from [13]. The system consists of three vessels, two continuously stirred tank reactors, and one tank separator. Both reactors are engaged in two reactions. The primary reaction involves converting reactant A into the main product B, while the secondary reaction leads to the undesired conversion of product A into the side-product C.

The considered state variables are temperatures T_i and concentrations of the reactant A, and products B and C, i.e., $c_{\mathrm{A},i}$, $c_{\mathrm{B},i}$, $c_{\mathrm{C},i}$ for the respective i-th vessel. The state variables are defined in the following order: $s = [T_1, c_{\mathrm{A},1}, c_{\mathrm{B},1}, c_{\mathrm{C},1}, T_2, c_{\mathrm{A},2}, c_{\mathrm{B},2}, c_{\mathrm{C},2}, T_3, c_{\mathrm{A},3}, c_{\mathrm{B},3}, c_{\mathrm{C},3}]^\top$. The control inputs are the heat flows H_1, H_2, and H_3, representing the external heat delivered or removed from the tanks. The detailed description of

the reactor-separator system can be found in [13]. The linearized state-space matrices along with all the data necessary to design the tube MPC can be found in Appendix A.

It is important to note, that the original (set-based) tube MPC was not possible to design due to extensive numerical demands related to geometric operations. On the contrary, it was possible to design and implement the implicit tube MPC. The closed-loop control simulation results are depicted in Figure 4.9 and Figure 4.10 for the system states and inputs, respectively. To conclude, all system states were successfully driven to the desired steady state in the presence of randomly generated disturbance, respecting the constraints.

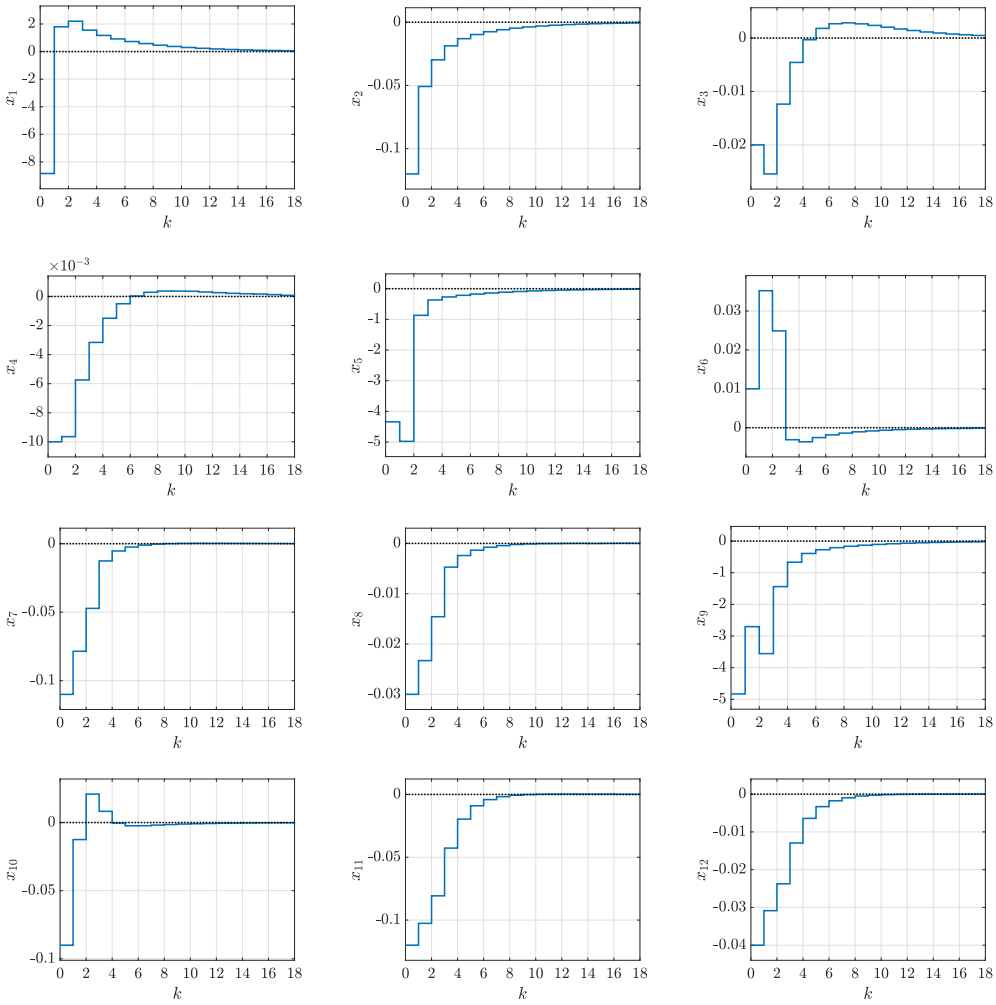


Figure 4.9: Closed-loop simulation of reactor-separator control – state variables. The solid lines represent the respective states and the dotted lines depict the target steady state in deviation form.

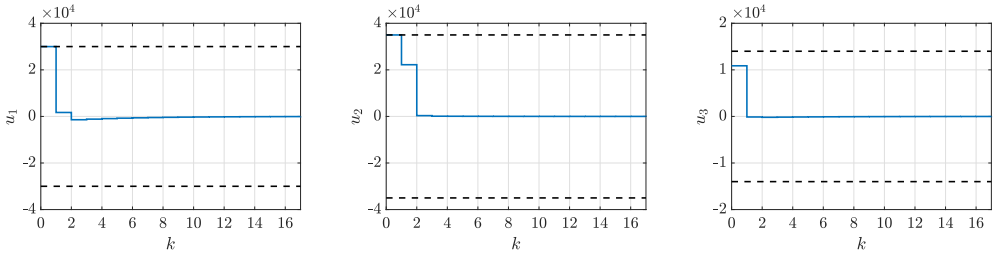


Figure 4.10: Closed-loop simulation of reactor-separator control – input variables. The solid lines represent the respective inputs and the dashed lines depict the constraints.

Chapter 5

Conclusions

This dissertation thesis deals with linear MPC design for complex systems. Throughout the thesis, the term "complex system" includes the systems that bring two groups of challenges: large-scale systems and systems with nonlinear and asymmetric dynamics. Three techniques of linear MPC were addressed: (i) partial explicit MPC, (ii) tunable approximated explicit MPC, and (iii) implicit tube MPC.

The first part of this thesis focuses on the topic of partial explicit model predictive control, which is suitable for large-scale problems with numerous constraints and optimization variables [32]. In the offline phase, only a subset of all critical regions is constructed and stored. In the online phase, the partial solution is utilized in the hot start strategy to streamline searching for the critical region where the measurement belongs and subsequent determination of the optimal control action. This strategy is a valuable tool for finding the solution of the MPC problem in high dimensions when solving the optimization problem online or developing the full solution offline is impractical and computationally demanding. Therefore, the partial explicit solution represents a compromise between the explicit MPC without any real-time optimization and computationally demanding implicit MPC.

One of the objectives was to propose novel ideas leading to memory footprint reduction. The main idea is to replace the polytopic critical region with its maximal volume inner approximation using the Chebyshev ball. As a consequence, the large-scale matrices defining the critical regions are no longer stored. Only the Chebyshev balls radii and centers are necessary to identify the nearest critical region. Then, the polytope is recovered from the optimization problem matrices and optimal active set. The rest of the online procedure remains the same.

Moreover, owing to the information about the radii of the Chebyshev balls, the determination of the nearest critical region becomes more accurate compared to the original approach as the inner approximation size is considered. Last but not least,

another valuable benefit of the proposed ideas is fixing the size of the partial solution memory footprint. The fixed structure of the Chebyshev ball data enables us to determine the partial solution memory footprint in advance, without the necessity to solve the large-scale optimization problem. As a result, the size of the partial explicit solution can be scaled to the control hardware memory limitations.

To support the theoretical background of the suggested concepts, 5 sets of large-scale MPC problems were randomly generated, and the associated memory footprints were evaluated. For every MPC problem, the partial explicit solution consisting of 300 critical regions was constructed. The average memory footprint of the partial solution based on the polytopic critical regions was evaluated for every problem size. Afterward, the memory footprint of the fixed-memory approach was determined based on the problem size. Analogously, the average memory footprint of the partial solution based on the Chebyshev balls was evaluated for every problem size. With increasing problem size, the contribution of the large-scale problem matrices significantly increases, and the gap between the total memory footprints of the two methods is lower. Therefore, also the memory savings decrease with the growing problem size. For the largest problem size consisting of 7 200 constraints, the total memory savings were 26.7 % and for the smallest problem size consisting of 6 030 constraints, the total memory savings were 41.2 %.

The second part of this dissertation thesis deals with the self-tunable explicit MPC design. This technique arises from the concept of tunable explicit MPC [36], which calculates the control action based on linear interpolation between the control actions from two boundary controllers. The boundary controllers differ only in the setup of one penalty matrix, and the rest of the structure and parameter values remain the same. For the price of suboptimality and storing two explicit controllers, one obtains the possibility to apply a control action that corresponds to an arbitrary controller setup chosen from the range between the lower and upper boundary controller setup.

The aim was to remove the necessity of manual interventions in the real-time control and delegate the real-time decision-making in tuning on well-developed internal rules, leading to an effective self-tuning technique for practical industry-oriented control applications. In numerous practical applications, the reference value of the controlled variable is changed and acquires values from a wide range of operating conditions. The use of different controller setups can help handle the plant's nonlinear and asymmetric behavior. This thesis presents two methods of the self-tunable controller technique. In the first approach, the controller's aggressivity is tuned based on the difference between the reference value and the steady state corresponding to the model linearization point. The technique represented a way how to compensate for the system's nonlinear

behavior. The second tuning technique was similar, taking into account the size of the reference change. The tuning parameter value is calculated as the ratio between the size of the reference change and the maximal admissible size of the reference change, which is specified before operation. Moreover, the tuning factor is scaled by splitting the interval of the tuning parameter into two ranges, which are assigned to different operating conditions, e.g., either positivity or negativity of the reference change. Following these rules, the asymmetry of the process dynamics is considered.

The following aim was to implement and analyze the control performance when using the self-tunable explicit MPC in a control of a laboratory heat exchanger. The case study included tracking a reference changing its value upwards and downwards to examine the proposed tuning method. Therefore, the tuning parameter interval was split into 2 parts. When the reference changed upwards, the control input was tuned in the first part of the interval and approached the boundary controller associated with the lower bound on the selected penalty matrix. On the contrary, when the reference changed downwards, the control input was tuned to approach the control input from the boundary controller with the upper bound on the penalty matrix.

To properly investigate the control results, the control performance was judged quantitatively using a set of quality criteria. Based on the evaluated criteria, the self-tunable control approach outperformed the conventional control strategy, handling just a single non-tunable controller. In average, the control performance criteria improved compared to the upper and lower boundary MPC respectively as follows: the squared-error-based criterion reduced by 23 % and 83 %, the maximal overshoot/undershoot reduced by 64 % and 102 %, and the settling time reduced by 12 % and 244 %.

The third part of the thesis is devoted to the topic of implicit tube MPC, which is suitable for controlling uncertain large-scale systems. When considering a large-scale system, the original tube MPC is not a suitable control technique, as it requires non-trivial geometric set operations, which is challenging in higher state-space dimensions [71]. A novel approach – implicit tube MPC – handles the MPC optimization problem differently, avoiding the construction of sets but still maintaining the optimal solution [67]. The aim of this thesis was to incorporate the implicit tube MPC into the MATLAB Multi-Parametric Toolbox extension, MPT+, to enable a wide usage and user-friendly design and implementation of this control technique. This thesis provides an overview of the tube MPC and implicit tube MPC design procedure in MPT+. The tube MPC was successfully implemented in a real-time control of Flexy^2 device using the MPT+ toolbox. Moreover, the validation of the implicit tube MPC incorporated in the toolbox is presented in a numerical control simulation. The implicit tube MPC was implemented on a large-scale reactor-separator system with 12

states and 3 control inputs. It is important to emphasize that the original (set-based) tube MPC was impossible to design due to extensive numerical demands related to geometric operations. On the contrary, it was possible to design and implement the implicit tube MPC, and all system states were successfully driven to the desired steady state in the presence of a randomly generated disturbance.

Main Contributions

The main contributions of this dissertation thesis are summarized in this section supported by the publications they were published in.

- **Theoretical contributions:**
  - **Memory footprint reduction associated with large-scale MPC.**
    The partial explicit MPC was revisited in order to analyze the memory footprint of the controller data. The size of the memory footprint to store the partial explicit MPC was reduced without inducing suboptimality of the applied control input. The memory reduction approach was published in:

    L. Galčíková – J. Oravec: Fixed-Complexity Solution of Partial Explicit MPC. Computers & Chemical Engineering, vol. 157, pp. 107606, 2022.

  - **Fixed size of memory footprint associated with large-scale MPC.**
    Besides the reduction of the memory footprint necessary to store the partial explicit MPC data, fixing the memory footprint size of the controller is addressed by considering the Chebyshev balls instead of the polytopic regions. The proposed method is published in:

    L. Galčíková – J. Oravec: Fixed-Complexity Solution of Partial Explicit MPC. Computers & Chemical Engineering, vol. 157, pp. 107606, 2022.

  - **Elaboration of the self-tuning technique for tunable explicit MPC to improve control performance.**
    The tunable approximated explicit MPC was revisited. The techniques to tune the explicit MPC automatically during the real-time control were elaborated, considering nonlinear and asymmetric plant behavior. The tuning technique based on the reference value was proposed in:

    L. Galčíková – M. Horváthová – J. Oravec – M. Bakošová: Self-Tunable Approximated Explicit Model Predictive Control of a Heat Exchanger. Chemical Engineering Transactions, vol. 94, pp. 1015–1020, 2022.

    The tuning technique, addressing also asymmetric plant behavior, was proposed in Chapter 3 of this thesis and is a part of the paper:

L. Galčíková – J. Oravec: Self-tunable approximated explicit MPC: Heat exchanger implementation and analysis. Journal of Process Control, 2023 (under review, round 2).

The preprint under review is available online:

`https://www.uiam.sk/assets/publication_info.php?id_pub=2621`

- **Practical validation:**
  - **Numerical case studies on large-scale systems.**
    The proposed partial explicit MPC memory reduction technique is analyzed on the set of large-scale systems. The memory footprint analysis was published in:

    L. Galčíková – J. Oravec: Fixed-Complexity Solution of Partial Explicit MPC. Computers & Chemical Engineering, vol. 157, pp. 107606, 2022.

  - **Laboratory validation on a heat exchanger plant.**
    The proposed self-tuning strategy was implemented on a laboratory heat exchanger. The control performance is analyzed and compared to non-tuned controllers. The experimental laboratory validation is a part of Chapter 3 of this thesis and of the paper:

    L. Galčíková – J. Oravec: Self-tunable approximated explicit MPC: Heat exchanger implementation and analysis. Journal of Process Control, 2023 (under review, round 2).

    The preprint under review is available online:

    `https://www.uiam.sk/assets/publication_info.php?id_pub=2621`

- **Freely available software:**
  - **Software development for control of large-scale systems.**
    The recent perspective approach of implicit tube MPC, suitable for controlling large-scale systems, was incorporated into the MATLAB MPT+ toolbox to spread the wide usage of this robust control technique. The toolbox was described in Chapter 4 of this thesis and is freely available online on GitHub:

    `https://github.com/holaza/mptplus`

– **Case study validating the developed software.**

The ancestor of the implicit tube MPC, i.e., the original set-based tube MPC, also incorporated to MPT+ toolbox, was validated on a laboratory device Flexy^2 [31] and published in:

J. Holaza – L. Galčíková – J. Oravec – M. Kvasnica: A software package for MPC design and tuning: MPT+. In 62nd IEEE Conference on Decision and Control, IEEE, Singapore, pp. 5682–5689, 2023.

The developed MPT+ extension enabling the implicit tube MPC application is validated on a large-scale reactor-separator system in Chapter 4 of this thesis.

Appendix A

Reactor-separator system

This Appendix section is devoted to the description of the reactor-separator system from Section 4.3 from the viewpoint of tube MPC design. The system is adapted from [13], where all, more specific parameters can be found, if necessary. The continuous-time state-space matrices of the reactor-separator system are stated at the end of this section due to large size. The system was discretized with a sampling time $T_\mathrm{s} = 0.1\,\mathrm{h}$. The steady state values s^s are defined as follows:

$$
\begin{aligned}
T_1^\mathrm{s} &= 369.53\,\mathrm{K}\\
c_{\mathrm{A},1}^\mathrm{s} &= 3.31\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{B},1}^\mathrm{s} &= 0.17\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{C},1}^\mathrm{s} &= 0.04\,\mathrm{kmol\,m^{-3}}\\
T_2^\mathrm{s} &= 435.25\,\mathrm{K}\\
c_{\mathrm{A},2}^\mathrm{s} &= 2.75\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{B},2}^\mathrm{s} &= 0.45\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{C},2}^\mathrm{s} &= 0.11\,\mathrm{kmol\,m^{-3}}\\
T_3^\mathrm{s} &= 435.25\,\mathrm{K}\\
c_{\mathrm{A},3}^\mathrm{s} &= 2.88\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{B},3}^\mathrm{s} &= 0.50\,\mathrm{kmol\,m^{-3}}\\
c_{\mathrm{C},3}^\mathrm{s} &= 0.12\,\mathrm{kmol\,m^{-3}}
\end{aligned}
$$

The constraints on the state, input variables and disturbances are:

$$\begin{aligned} -s^{\mathrm{s}} &\preceq \quad x \quad \preceq 100 \cdot s^{\mathrm{s}}, \\ [-30,-35,-14]^{\top} \cdot 10^5 &\preceq \quad u \quad \preceq [30,35,14]^{\top} \cdot 10^5, \\ -0.001I &\preceq \quad d \quad \preceq 0.001I, \end{aligned}$$

where I denotes the identity vector of the corresponding dimensions. The matrix E multiplying the vector of disturbances d in (4.1) was set as identity matrix.

Regarding the MPC parameters, the prediction horizon N was 18 steps long and the penalty matrices were set as follows:

$$\begin{aligned} &Q = \mathrm{diag}(Q_1, Q_2, Q_3), \text{ where } Q_i = \mathrm{diag}(3200,1,1,1) \cdot 10^3, \ i = 1,2,3, \\ &R = \mathrm{diag}(1,1,1) \cdot 10^{-2}. \end{aligned}$$

The terminal penalty and terminal set was calculated using the option `LQRStability`=1.

Finally, the initial state condition for the performed control simulation in Section 4.3 was set as:

$$x_0 = [-8.84, -0.12, -0.02, -0.01, -4.34, 0.01, -0.11, -0.03, -4.83, -0.09, -0.12, -0.04]^{\top}.$$

$$
A_c = \left[\begin{array}{cccccccccccc}
3.049 & 67.090 & 0 & 0 & 0 & 0 & 0 & 0 & 1.900 & 0 & 0 & 0 \\
-0.046 & -7.207 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.203 & -0.068 & -0.102 \\
0.037 & 0.257 & -6.900 & 0 & 0 & 0 & 0 & 0 & 0 & -0.012 & 0.663 & -0.009 \\
0.008 & 0.050 & 0 & -6.900 & 0 & 0 & 0 & 0 & 0 & -0.004 & -0.002 & 1.001 \\
13.766 & 0 & 0 & 0 & 49.396 & 820.799 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 13.766 & 0 & 0 & -0.334 & -27.513 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 13.766 & 0 & 0.261 & 2.995 & -23.766 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 13.766 & 0.072 & 0.752 & 0 & -23.766 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 12.440 & 0 & 0 & 0 & -12.440 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 12.440 & 0 & 0 & 0 & -11.743 & 0.068 & 0.102 \\
0 & 0 & 0 & 0 & 0 & 0 & 12.440 & 0 & 0 & 0.012 & -11.203 & 0.009 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 12.440 & 0 & 0.004 & 0.002 & -11.541
\end{array}\right]
$$

$$
B_c = \left[\begin{array}{cccccc}
0.0043 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.0087 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.0043 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{array}\right]
$$

Appendix B

Author's publications

Papers in journal

- **L. Galčíková** – J. Oravec: Self-tunable approximated explicit MPC: Heat exchanger implementation and analysis. Journal of Process Control, 2023 (under review, round 2).
- P. Bakaráč – M. Horváthová – **L. Galčíková** – J. Oravec – M. Bakošová: Approximated MPC for embedded hardware: Recursive random shooting approach. Computers & Chemical Engineering, vol. 165, 2022.
- **L. Galčíková** – M. Horváthová – J. Oravec – M. Bakošová: Self-tunable approximated explicit model predictive control of a heat exchanger. Chemical Engineering Transactions, vol. 94, pp. 1015–1020, 2022.
- **L. Galčíková** – J. Oravec: Fixed-complexity solution of partial explicit MPC. Computers & Chemical Engineering, vol. 157, pp. 107606, 2022.
- J. Oravec – M. Bakošová – **L. Galčíková** – M. Slávik – M. Horváthová – A. Mészáros: Soft-constrained robust model predictive control of a plate heat exchanger: Experimental analysis. Energy, vol. 180, pp. 303–314, 2019.
- J. Oravec – M. Bakošová – M. Horváthová – **L. Galčíková** – M. Slávik – A. Vasičkaninová – A. Mészáros: Convex-lifting-based robust control of a laboratory plate heat exchanger. Chemical Engineering Transactions, vol. 81, pp. 733–738, 2019.

Papers in conference proceedings

- **L. Galčíková** – P. Belková – J. Oravec: Real-time tuning of approximated explicit MPC of a heat exchanger. In Proceedings of the 24th International Conference on Process Control - Summaries Volume, Štrbské Pleso, Slovakia, 2021 (presented in person).
- J. Holaza – **L. Galčíková** – J. Oravec – M. Kvasnica: A software package for MPC design and tuning: MPT+. In 62nd IEEE Conference on Decision and Control, IEEE, Singapore, pp. 5682–5689, 2023 (presented in person).
- M. Horváthová – **L. Galčíková** – M. Klaučo – J. Oravec: Real-Time Optimisation-Based Robust Control: Heat Exchanger Comparative Analysis. In 62nd IEEE Conference on Decision and Control, IEEE, Singapore, pp. 6535–6540, 2023.
- M. Horváthová – **L. Galčíková** – J. Oravec: Control Design for a Nonlinear Reactors-Separator Plant. In 2022 Cybernetics & Informatics (K&I), pp. 1–6, 2022.
- R. Kohút – **L. Galčíková** – K. Fedorová – T. Ábelová – M. Bakošová – M. Kvasnica: Hidden Markov Model-based Warm-start of Active Set Method in Model Predictive Control. In Proceedings of the 23rd International Conference on Process Control, Štrbské Pleso, Slovakia, 2021.
- **L. Galčíková** – J. Oravec – M. Bakošová: Advanced Robust MPC Design for Plate Heat Exchanger. In Proceedings of the 22nd International Conference on Process Control, Slovak Chemical Library, Štrbské Pleso, Slovakia, 2019.

Appendix C

Curriculum vitae

Lenka Galčíková

Email:	lenka.galcikova@stuba.sk
Homepage:	`https://www.uiam.sk/~galcikova/`
ORCID iD:	0000-0002-2213-6799
Year of Birth:	1995

Education

- 2020 – present PhD-study: Faculty of Chemical and Food Technology, STU in Bratislava, Study Programme: Process Control
- 2018 – 2020 Master-study: Faculty of Chemical and Food Technology, STU in Bratislava, Study Programme: Automation and Information Engineering in Chemistry and Food Industry
- 2015 – 2018 Bachelor-study: Faculty of Chemical and Food Technology, STU in Bratislava, Study Programme: Automation, Information Engineering and Management in Chemistry and Food Industry

Participation on research projects

European Projects

- 2022 – 2025 Fostering Opportunities Towards Slovak Excellence in Advanced Control for Smart Industries (Investigator)

APVV Research Projects

- 2021 – 2024 Energy-efficient Safe and Secure Process Control (Investigator)

VEGA Research Projects

- 2022 – 2025 Controller design methods for low-level carbon footprint process automation (Investigator)
- 2021 – 2023 Aggregation of uncertain data represented by intervals and vectors (Investigator)
- 2020 – 2023 Advanced Control of Energy Intensive Processes with Uncertainties in Chemical, Biochemical and Food Technologies (Investigator)

Other Projects

- 2022 – 2023 Computationally effective optimal control of plants in chemical industry (Internal STU scheme, principal investigator)
- 2021 – 2022 Complexity reduction of explicit model predictive control of plants in chemical industry (Internal STU scheme, principal investigator)
- 2021 – 2023 Design of Optimal Controllers for Industrial Microprocessors (Internal STU scheme, investigator)
- 2021 – 2022 Construction of a Smart Eco Greenhouse VESNA (Tatra bank foundation, investigator)

International visits and study stays

- 2022 – EECI International Graduate School on Control – Dynamics and Algorithms on Networks, Paris Saclay, France
- 2021 – Chinese MPC School, online

Awards

- 2023 Dean's Praise for outstanding fulfilment of academic duties and activities for the benefit of the Faculty
- 2022 Dean's Praise for activities for the benefit of the Faculty
- 2020 Award of the Dean of FCHPT in Bratislava for excellent results during the graduate studies
- 2020 Award of the Rector of STU in Bratislava for excellent performance of study obligations throughout the sudies of Engineer's study programme
- 2020 Student of the year 2020 in the category Outstanding activity performed for the development or promotion of STU in Bratislava
- 2019 Student of the year 2019 in the category Outstanding study results (2nd level) at FCHPT STU in Bratislava
- 2019 Dean's Praise for outstanding fulfilment of academic duties, extraordinary results in research and activities for the benefit of the Faculty
- 2019 Winner of the Female Engineers MOL Challenge `https://femp.molgroup.info/en/`

Teaching activities

- Automatic Control Theory I (Master study, laboratory exercises)
- Automatic Control Theory III (Master study, laboratory exercises)
- Semmestral Project I-III (Master study, laboratory exercises)
- Fundamentals of MATLAB (Bachelor study, laboratory exercises)
- Laboratory Exercises of Process Control (Bachelor study, laboratory exercises)

Community engagement

- 2020 – 2023 International Student Scientific Conference at FCHPT STU in Bratislava, vice-chair of the organizing comitee

Appendix D

Resumé

Táto dizertačná práca sa zaoberá návrhom lineárneho prediktívneho riadenia pre zložité systémy. Pojem "zložitý systém" v celej práci označuje systémy, ktoré prinášajú dve skupiny výziev pri návrhu riadenia: veľkorozmerné systémy a systémy s nelineárnou a asymetrickou dynamikou. Táto práca sa zameriava na tri techniky lineárneho MPC: (i) parciálne explicitné MPC, (ii) laditeľné aproximované explicitné MPC a (iii) MPC založené na implicitných tubách.

Prvá časť práce je zameraná na tému parciálneho explicitného MPC, ktoré je vhodné pre veľkorozmerné problémy s veľkým množstvom ohraničení a optimalizovaných premenných [32]. Najskôr sa v offline fáze vytvorí a uloží iba podmnožina všetkých kritických regiónov. Následne sa v online fáze toto parciálne riešenie využíva v inicializačnej stratégii, tzv. hot-start stratégii, na zefektívnenie hľadania kritického regiónu, kam patrí meranie a následné určenie optimálneho akčného zásahu. Táto stratégia je cenným nástrojom pre riešenie MPC optimalizačného problému vo veľkých rozmeroch, keď je riešenie optimalizačného problému online alebo zostrojenie úplného explicitného riešenia offline nepraktické a výpočtovo náročné. Čiastočné explicitné riešenie preto predstavuje kompromis medzi explicitným MPC, ktoré nevyžaduje optimalizáciu počas riadenia, a výpočtovo náročným implicitným MPC.

Jedným z cieľov práce bolo navrhnúť nové metódy vedúce k zníženiu pamäťovej stopy MPC regulátora. Hlavnou myšlienkou bolo nahradiť polytopický kritický región jeho vnútornou aproximáciou založenou na kruhoch s maximálnym objemom. V dôsledku toho sa už neukladajú veľkorozmerné matice definujúce kritické regióny. Na identifikáciu najbližšieho kritického regiónu sú potrebné iba polomery a stredy vpísaných kruhov. Potom sa príslušný kritický región získa z matíc optimalizačného problému MPC a optimálnej množiny aktívnych ohraničení. Zvyšok online fázy zostáva rovnaký ako v pôvodnom navrhnutom postupe.

Navyše, vďaka informácii o polomere vpísaných kruhov je určovanie najbližších kri-

tických regiónov presnejšie, keďže sa berie do úvahy veľkosť vnútornej aproximácie. V neposlednom rade, ďalšou významnou výhodou navrhnutej metódy je zafixovanie veľkosti pamäte parciálneho riešenia. Fixná štruktúra vpísaného kruhu nám umožňuje určiť veľkosť pamäte parciálneho riešenia vopred, bez potreby vyriešiť veľkorozmerný optimalizačný problém.

Vlastnosti navrhnutej metódy boli analyzované na numerickom príklade. Najskôr bolo vytvorených 5 sád veľkorozmerných MPC problémov o rôznej veľkosti. Pre každý MPC problém bolo zostrojené parciálne explicitné riešenie pozostávajúce z 300 kritických regiónov. Pre každú veľkosť problému bola vyhodnotená priemerná veľkosť pamäte parciálneho riešenia na základe prístupu založenom na polytopických kritických regiónoch. Následne bola vyhodnotená aj veľkosť pamäťovej stopy založená na prístupe fixovanej zložitosti. Analogicky sa potom vyhodnotila priemerná veľkosť pamäte pre každú veľkosť problému. S rastúcou veľkosťou problému sa výrazne zvyšuje príspevok rozsiahlych veľkorozmerných matíc optimalizačného problému MPC, v dôsledku čoho je rozdiel medzi celkovými pamäťovými stopami týchto dvoch metód nižší. Preto aj úspora pamäte klesá s rastúcou veľkosťou problému. Pre najväčšiu veľkosť problému pozostávajúcu zo 7 200 ohraničení bola celková úspora pamäte 26,7 % a pre najmenšiu veľkosť problému pozostávajúcu zo 6 030 ohraničení bola celková úspora pamäte 41,2 %.

Ak sa navrhované techniky zapracujú do pôvodného prístupu parciálneho explicitného prediktíveho riadenia, vedú k významným pamäťovým úsporám, ktoré sú podstatné pre ukladanie veľkorozmerných dát. Ďalším cenným príspevkom je fixovanie a určenie veľkosti vopred, bez potreby vyriešenia optimalizačného problému.

Druhá časť tejto dizertačnej práce sa zaoberá návrhom metódy samoladiteľného explicitného MPC. Táto technika vychádza z konceptu laditeľného explicitného MPC [36], ktoré počíta akčný zásah na základe lineárnej interpolácie medzi akčnými zásahmmi z dvoch hraničných regulátorov. Tieto hraničné regulátory sa líšia iba nastavením jednej penalizačnej matice a zvyšok štruktúry reulátora a hodnôt parametrov zostáva rovnaký. Za cenu suboptimálnosti a uloženia dvoch hraničných explicitných regulátorov sa získa možnosť aplikovať akčný zásah, ktorý zodpovedá aproximácii ľubovoľného nastavenia regulátora zvoleného z rozsahu medzi nastavením dolného a horného hraničného regulátora.

Cieľom práce bolo odstrániť nutnosť manuálnych zásahov do riadenia v reálnom čase a delegovať rozhodovanie v reálnom čase pri ladení na dobre vypracované interné pravidlá, vedúce k efektívnej technike samoladenia pre praktické priemyselné aplikácie riadenia. V mnohých praktických aplikáciách sa referenčná hodnota regulovanej veličiny mení a nadobúda hodnoty zo širokého spektra možných prevádzkových podmienok.

Použitie rôznych nastavení regulátora môže pomôcť zvládnuť nelineárne a asymetrické správanie zariadenia. Táto práca predstavuje dve techniky nastavenia samoladiteľného regulátora. V prvom prístupe je agresivita regulátora ladená na základe rozdielu medzi referenčnou hodnotou a ustáleným stavom zodpovedajúcim bodu linearizácie modelu. Táto technika predstavovuje spôsob, ako kompenzovať nelineárne správanie systému. Druhá technika ladenia bola podobná, berúc do úvahy veľkosť zmeny referencie. Hodnota ladiaceho parametra sa vypočíta ako pomer medzi veľkosťou zmeny referencie a maximálnou prípustnou veľkosťou zmeny referencie, ktorá je špecifikovaná pred samotnou prevádzkou. Okrem toho je ladiaci faktor následne škálovaný rozdelením intervalu ladiaceho parametra na dva rozsahy, ktoré sú priradené rôznym prevádzkovým podmienkam, napr. kladnej alebo zápornej zmene hodnoty referencie. Nastavením týchto pravidiel je možné zohľadniť asymetriu dynamiky procesu.

Nasledujúcim cieľom bolo implementovať a analyzovať kvalitu riadenia s použitím samoladiteľného explicitného MPC regulátora pri riadení laboratórneho výmenníka tepla. Prípadová štúdia zahŕňala úlohu sledovania referencie, ktorá mení svoju hodnotu smerom nahor a nadol, aby sa preskúmala navrhovaná metóda ladenia. Preto bol interval parametrov ladenia rozdelený na 2 časti. Keď sa referencia zmenila smerom nahor, akčný zásah sa naladil v prvej časti intervalu a priblížil sa k akčným zásahom dolného hraničného regulátora. Naopak, keď sa referencia zmenila smerom nadol, akčný zásah bol naladený tak, aby dal väčšiu váhu akčného zásahu z horného hraničného regulátora.

V rámci hlbšieho preskúmania výsledkov riadenia bol kvantitatívne vyhodnotený súbor kritérií kvality. Na základe vyhodnotených kritérií možno konštatovať, že prístup samoladiteľného riadenia prekonal konvenčnú stratégiu riadenia založenú na jednom explicitnom MPC regulátore. V priemere sa výkonnostné kritériá kvality zlepšili v porovnaní s horným a dolným hraničným MPC, v tomto poradí, nasledovne: kritérium na základe sumy štvorcov regulačnej odchýlky sa znížilo o 23 % a 83 %, maximálny prekmit/podkmit sa znížil o 64 ,% a 102 % a čas regulácie sa skrátil o 12 % a 244 %.

Tretia časť práce je venovaná téme MPC založeného na implicitných tubách, ktoré je vhodné na riadenie neurčitých veľkorozmerných systémov. Pôvodné tube MPC nie je vhodnou riadiacou technikou pre veľkorozmerné systémy, pretože vyžaduje netriviálne množinové operácie, čo je náročné vo vyšších rozmeroch stavov [71]. Nový prístup – MPC založené na implicitných tubách – navrhuje optimalizačný problém MPC odlišne, vynecháva konštrukciu geometrických množín, ale stále zachováva optimálne riešenie [67]. Cieľom tejto práce bolo začleniť MPC založené na implicitných tubách do MPT+ (z angl. Multi-Parammetric Toolbox) toolboxu v prostredí MATLAB, aby sa umožnilo široké využitie a užívateľsky priateľský návrh a implementácia tejto

techniky riadenia. Táto práca poskytuje prehľad postupu návrhu MPC založeného na tubách a implicitných tubách v MPT+. MPC založené na tubách bolo úspešne implementované pri riadení zariadenia Flexy^2 v reálnom čase pomocou toolboxu MPT+. Navyše, validácia MPC založeného na implicitných tubách začleneného do toolboxu je prezentovaná v simulácii riadenia. MPC založené na implicitných tubách bolo implementované na veľkorozmernom systéme dvoch reaktorov a separátora s 12 stavmi a 3 riadiacimi vstupmi. Je dôležité zdôrazniť, že pôvodné MPC založené na geometrických tubách nebolo možné navrhnúť z numerických dôvodov súvisiacimi s geometrickými operáciami. Naopak, bolo možné navrhnúť a implementovať MPC založené na implicitných tubách a všetky stavy systému boli úspešne riadené do požadovaného ustáleného stavu v prítomnosti náhodne generovanej poruchy.

Bibliography

[1] *Introduction Manual, PCT23-MkII Process Plant Trainer*. Armfield, 2007.

[2] MPT+ Toolbox, 2024. `https://github.com/holaza/mptplus`.

[3] J.A.E. Andersson, J. Gillis, G. Horn, J.B. Rawlings, and M. Diehl. CasADi – a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[4] M. Baric, M. Baotic, and M. Morari. On-line tuning of controllers for systems with constraints. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 8288–8293, 2005.

[5] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38:3–20, 2002.

[6] F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

[7] F. Borrelli. *Constrained Optimal Control of Linear and Hybrid Systems*. Springer Berlin Heidelberg, 2017.

[8] F. Borrelli, A. Bemporad, and M. Morari. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[9] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[10] B. Burnak, N. A. Diangelakis, J. Katz, and E. N. Pistikopoulos. Integrated process design, scheduling, and control using multiparametric programming. *Computers & Chemical Engineering*, 125:164–184, 2019.

[11] Y. Chen, M. Bruschetta, E. Picotti, and A. Beghi. MATMPC - A MATLAB based toolbox for real-time nonlinear model predictive control. *CoRR*, abs/1811.08761, 2018.

[12] Y. Chen, R. Dwivedi, M. Wainwright, and B. Yu. Fast mcmc sampling algorithms on polytopes. *Journal of Machine Learning Research*, 19, 10 2017.

[13] P. D. Christofides, Jinfeng Liu, and David Muñoz de la Peña. *Multirate Distributed Model Predictive Control*, pages 193–218. Springer London, London, 2011.

[14] M. L. Darby and M. Nikolaou. MPC: Current practice and challenges. *Control Engineering Practice*, 20(4):328–342, 2012.

[15] M. L. Darby, M. Nikolaou, J. Jones, and D. Nicholson. RTO: An overview and assessment of current practice. *Journal of Process Control*, 21(6):874–884, 2011.

[16] V. Dua, N. A. Bozinis, and E. N. Pistikopoulos. A multiparametric programming approach for mixed-integer quadratic engineering problems. *Computers & Chemical Engineering*, 26(4):715–733, 2002.

[17] H. J. Ferreau, H. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18:816–830, 2008.

[18] A. V. Fiacco. *Introduction to sensitivity and stability analysis in nonlinear programming*, volume 165. Academic press, 1983.

[19] L. Galčíková, M. Horváthová, J. Oravec, and M. Bakošová. Self-tunable approximated explicit model predictive control of a heat exchanger. *Chemical Engineering Transactions, 2022, Vol. 94*, 94:1015–1020, 2022.

[20] G. Goebel and F. Allgöwer. Semi-explicit MPC based on subspace clustering. *Automatica*, 83:309–316, 2017.

[21] A. Gupta, S. Bhartiya, and P. S. V. Nataraj. A novel approach to multiparametric quadratic programming. *Automatica*, 47(9):2112–2117, 2011.

[22] M. Herceg, M. Kvasnica, C. Jones, and M. Morari. Multi-parametric toolbox 3.0. In *2013 European Control Conference*, pages 502–510, 2013.

[23] M. Herceg, M. Kvasnica, C. Jones, and M. Morari. Multi-parametric toolbox 3.0. In *European Control Conference*, pages 502–510, 2013.

[24] J. Holaza, P. Bakaráč, and J. Oravec. Revisiting reachability-driven explicit MPC for embedded control. *European Journal of Control*, page 101019, 2024.

[25] J. Holaza, L. Galčíková, J. Oravec, and M. Kvasnica. A software package for MPC design and tuning: MPT+. In *62nd IEEE Conference on Decision and Control*, pages 5682–5689, Singapore, December 13-15 2023. IEEE.

[26] J. Holaza, K. Kvasnicová, E. Pavlovičová, and J. Oravec. Tube MPC extension of MPT: Experimental analysis. In R. Paulen and M. Fikar, editors, *Proceedings of the 2023 24th International Conference on Process Control*, pages 120–125. Slovak University of Technology in Bratislava, IEEE, June 6 - 9, 2023 2023.

[27] J. Holaza, J. Oravec, M. Kvasnica, R. Dyrska, M. Mönnigmann, and M. Fikar. Accelerating explicit model predictive control by constraint sorting. In *Preprints of the 21st IFAC World Congress (Virtual), Berlin, Germany*, volume 21, pages 11520–11525, 2020.

[28] B. Houska, H.J. Ferreau, and M. Diehl. ACADO Toolkit – an open source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[29] B. Houska and J. Shi. Distributed MPC with ALADIN–a tutorial. In *2022 American Control Conference (ACC)*, pages 358–363, 2022.

[30] M. Jost, G. Pannocchia, and M. Mönnigmann. Accelerating linear model predictive control by constraint removal. *European Journal of Control*, 35:42–49, 2017.

[31] M. Kalúz, M. Klaučo, Ľ. Čirka, and M. Fikar. Flexy2: A portable laboratory device for control engineering education. *IFAC-PapersOnLine*, 52(9):42–47, 2019. 12th IFAC Symposium on Advances in Control Education.

[32] J. Katz and E. N. Pistikopoulos. A partial multiparametric optimization strategy to improve the computational performance of model predictive control. *Computers & Chemical Engineering*, 118:107057, 2020.

[33] K. Kiš, P. Bakaráč, and M. Klaučo. Nearly optimal tunable MPC strategies on embedded platforms. In *18th IFAC Workshop on Control Applications of Optimization*, pages 326–331. IFAC-PapersOnline, 2022.

[34] K. Kiš, M. Klaučo, and A. Mészáros. Neural network controllers in chemical technologies. In *2020 IEEE 15th International Conference of System of Systems Engineering (SoSE)*, pages 397–402, 2020.

[35] M. Klaučo and M. Kvasnica. *MPC-Based Reference Governors*. Springer, 2019.

[36] M. Klaučo and M. Kvasnica. Towards on-line tunable explicit MPC using interpolation. In *Preprints of the 6th IFAC Conference on Nonlinear Model Predictive Controle*, Madison, Wisconsin, USA, 2018. IFAC.

[37] D. Kouzoupis, G. Frison, A. Zanelli, and M. Diehl. Recent advances in quadratic programming algorithms for nonlinear model predictive control. *Vietnam J. Math*, 46:863–882, 2018.

[38] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. University of California Press, Berkeley and Los Angeles, 1951.

[39] M. Kvasnica, P. Bakaráč, and M. Klaučo. Complexity reduction in explicit MPC: A reachability approach. *Systems & Control Letters*, 124:19–26, 2019.

[40] M. Kvasnica and M. Fikar. Clipping-based complexity reduction in explicit mpc. *IEEE Transactions on Automatic Control*, 57(7):1878–1883, 2012.

[41] M. Kvasnica, J. Hledík, I. Rauová, and M. Fikar. Complexity reduction of explicit model predictive control via separation. *Automatica*, 49(6):1776–1781, 2013.

[42] M. Kvasnica, B. Takács, J. Holaza, and S. Di Cairano. On region-free explicit model predictive control. In *54th IEEE Conference on Decision and Control*, volume 54, pages 3669–3674, Osaka, Japan, December 15-18, 2015 2015.

[43] P. Lancaster and L. Rodman. *Algebraic Riccati equations*. Clarendon press, 1995.

[44] H. Li and C. L. E. Swartz. Dynamic real-time optimization of distributed MPC systems using rigorous closed-loop prediction. *Computers & Chemical Engineering*, 122:356–371, 2019. 2017 Edition of the European Symposium on Computer Aided Process Engineering (ESCAPE-27).

[45] D. Limon, I. Alvarado, T. Alamo, and E.F. Camacho. Robust tube-based MPC for tracking of constrained linear systems with additive disturbances. *Journal of Process Control*, 20(3):248–260, 2010.

[46] B. G. Liptak. *Instrument Engineers' Handbook*, volume 2: Process Control and Optimization. CRC Press, London, 4 edition, 2005.

[47] J. Löfberg. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *Proc. of the CACSD Conf.*, pages 284–289, Taipei, Taiwan, 2004.

[48] J. Z. Lu. Closing the gap between planning and control: A multiscale MPC cascade approach. *Annual Reviews in Control*, 40:3–13, 2015.

[49] S. Lucia, A. Tätulea-Codrean, C. Schoppmeyer, and S. Engell. Rapid development of modular and sustainable nonlinear model predictive control solutions. *Control Engineering Practice*, 60:51–62, 2017.

[50] J. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, London, 2000.

[51] D. Masti, T. Pippia, A. Bemporad, and B. De Schutter. Learning approximate semi-explicit hybrid MPC with an application to microgrids. *IFAC-PapersOnLine*, 53(2):5207–5212, 2020. 21th IFAC World Congress.

[52] MathWorks. MPC Toolbox, 2024. `https://www.mathworks.com/products/model-predictive-control.html`.

[53] D. Mayne. Robust and stochastic model predictive control: Are we going in the right direction? *Annual Reviews in Control*, 41:184–192, 2016.

[54] D. Q. Mayne, S. V. Raković, R. Findeisen, and F. Allgöwer. Robust output feedback model predictive control of constrained linear systems. *Automatica*, 42(7):1217–1222, 2006.

[55] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[56] D.Q. Mayne, M.M. Seron, and S. V. Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224, 2005.

[57] J. Mikleš and M. Fikar. *Process Modelling, Identification, and Control*. Springer, 2007.

[58] R. Mitze and M. Mönnigmann. A dynamic programming approach to solving constrained linear–quadratic optimal control problems. *Automatica*, 120:109132, 2020.

[59] M. Morari and J. H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(10):667–682, 1999.

[60] M. M. Morato, J. E. Normey-Rico, and Olivier Sename. Model predictive control design for linear parameter varying systems: A survey. *Annual Reviews in Control*, 49:64–80, 2020.

[61] M. Mönnigmann. On the structure of the set of active sets in constrained linear quadratic regulation. *Automatica*, 106:61–69, 2019.

[62] J. Oravec, S. Blažek, M. Kvasnica, and S. Di Cairano. Polygonic representation of explicit model predictive control. In *IEEE Conference on Decision and Control*, pages 6422–6427, Florence, Italy, 2013.

[63] J. Oravec and M. Klaučo. Real-time tunable approximated explicit MPC. *Automatica*, 142:110315, 2022.

[64] I. Pappas, D. Kenefake, B. Burnak, S. Avraamidou, H. S. Ganesh, J. Katz, N. A. Diangelakis, and E. N. Pistikopoulos. Multiparametric programming in process systems engineering: Recent developments and path forward. *Frontiers in Chemical Engineering*, 2, 2021.

[65] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[66] D. M. Raimondo, D. Limon, M. Lazar, L. Magni, and E. F. Camacho. Min-max model predictive control of nonlinear systems: A unifying overview on stability. *European Journal of Control*, 15(1):5–21, 2009.

[67] S. V. Raković. The implicit rigid tube model predictive control. *Automatica*, 157:111234, 2023.

[68] S. V. Raković, E. C. Kerrigan, K. I. Kouramas, and D. Q. Mayne. Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 50(3):406–410, 2005.

[69] S. V. Raković, B. Kouvaritakis, R. F., and M. Cannon. Homothetic tube model predictive control. *Automatica*, 48(8):1631–1638, 2012.

[70] S. V. Raković, William S. Levine, and Behçet Açikmese. Elastic tube model predictive control. In *2016 American Control Conference (ACC)*, pages 3594–3599, 2016.

[71] S. V. Raković and D.Q. Mayne. A simple tube controller for efficient robust model predictive control of constrained linear discrete time systems subject to bounded disturbances. *IFAC Proceedings Volumes*, 38(1):241–246, 2005. 16th IFAC World Congress.

[72] S.V. Raković and D.Q. Mayne. A simple tube controller for efficient robust model predictive control of constrained linear discrete time systems subject to bounded disturbances. *IFAC Proceedings Volumes*, 38(1):241–246, 2005. 16th IFAC World Congress.

[73] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. 2 edition, 2019.

[74] D. Di Ruscio. Model predictive control with integral action: A simple MPC algorithm. *Modeling, Identification and Control*, 34(3):119–129, 2013.

[75] P.O.M. Scokaert and D.Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43(8):1136–1142, 1998.

[76] R. L. Smith. The hit-and-run sampler: a globally reaching markov chain sampler for generating arbitrary multivariate distributions. In *Proceedings Winter Simulation Conference*, pages 260–264, 1996.

[77] S. H. Son, T. H. Oh, J. W. Kim, and J. M. Lee. Move blocked model predictive control with improved optimality using semi-explicit approach for applying time-varying blocking structure. *Journal of Process Control*, 92:50–61, 2020.

[78] J. Spjøtvold, E. C. Kerrigan, C. N. Jones, P. Tøndel, and T. A. Johansen. On the facet-to-facet property of solutions to convex parametric quadratic programs. *Automatica*, 42(12):2209–2214, 2006.

[79] J. Spjøtvold, S. V. Raković, P. Tøndel, and T. A. Johansen. Utilizing reachability analysis in point location problems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 4568–4569, 2006.

[80] G. P. H. Styan. Hadamard products and multivariate statistical analysis. *Linear Algebra and its Applications*, 6:217–240, 1973.

[81] P. Tøndel, T. A. Johansen, and A. Bemporad. An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, 39(3):489–497, 2003.

[82] K. Wang, Y. Jiang, J. Oravec, M. Villanueva, and B. Houska. Parallel explicit tube model predictive control. In *58th IEEE Conference on Decision and Control*, 58, pages 7696–7701, Nice, France, December 11-13, 2019 2019.

[83] M. N. Zeilinger, C. N. Jones, and M. Morari. Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization. *IEEE Transactions on Automatic Control*, 56(7):1524–1534, 2011.