NTNU – Trondheim
Norwegian University of
Science and Technology

Penalty Function Approaches for Proactive Fault-Tolerant Model Predictive Control

Jon Håman Brusevold

Master of Science in Cybernetics and Robotics
Submission date: June 2015
Supervisor: Bjarne Anton Foss, ITK
Co-supervisor: Brage Rugstad Knudsen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

NTNU
Norwegian University of Science and Technology

Faculty of Information Technology,
Mathematics and Electrical
Engineering

Department of Engineering Cybernetics

Master project

Name of candidate: Jon Håman Brusevold

Subject: Engineering Cybernetics

Title: On Penalty-Function Approaches For Fault-Taulerant Model-Predictive Control

Title (in Norwegian): Penalty funksjon tilnærminger for feil tolerant model prediktiv regulering

The main task for this master project is to explore penalty function formulations for proactive model predictive control (MPC) for handling incipient faults, in particular actuation faults. The project is based on the idea using soft constraints with a penalty function for fault-tolerant MPC with an economic criteria, so as to preserve the economics of a process during nominal operations, while driving the system into a safety set in the event of an incipient fault. The project should focus on linear time-invariant (LTI) systems.

Based on the project report entitled *Fault-tolerant MPC for active mitigation of actuation faults in process systems* by the same candidate, the master project should explore the following topics:

1. Explore computational strategies for computing penalty parameters for exact penalty functions with soft constraints for safety sets. In particular, implement and analyze an approach using bi-level programming for designing penalty function for MPC.
2. Briefly explore and describe general methods for computing control invariant sets, and describe how these sets enters into the FTMPC approach.
3. Consider and analyze stability properties of the proposed MPC approach.
4. Extend the proposed FTMPC scheme for nominal systems to be robust to disturbances.
5. Analyze and discuss criteria for optimal penalty formulations for safety sets with soft constraints. That is, consider trade-offs between preserving economic operations of the process and robustness of the controller by quickly driving the system into the stabilize region for the faulty system.
6. Consider limitations on achievable performance for the proposed approach. In particular, for LTI systems, consider approaches to compute and possibly constrain how far away the state can move from the safety set while still being able to reach this set within the allowable time (i.e., before the incipient fault is likely to occur).

The thesis report may include a draft paper to a selected conference with the main results.

Starting date: 12.01.2015

End date: 12.06.2015

Co-supervisor: Brage Rugstad Knudsen, Postdoc

Trondheim, 14.01.2015
Bjarne Foss
Professor/supervisor

Preface

This Master Thesis is written in the spring of 2015, and is the final result from my five-year study at the Master of Technology program in Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU).

I would like to thank my supervisor Professor Bjarne A. Foss for constructive and encouraging inputs and discussions. A special thank you goes to my co-supervisor Postdoc Researcher Brage R. Knudsen. I am extremely grateful for all the valuable inputs he has given me throughout my last year at NTNU. He has always been available for answering questions, and has taken an active interest in my master thesis. The development of the theory in this thesis is in collaboration with Brage, and would not have been possible without his guidance. I would also like to thank my fellow students in GG-44 for a great working environment, as well as valuable discussions during the course of this work.

The report includes a draft paper with the main results submitted to the IEEE *Conference on Decision and Control* 2015, and an additional paper is in the making.

Trondheim, 2015-06-10

Jon Håman Brusevold

Abstract

In the event of actuator faults in systems, standard control algorithms might not be sufficient for stabilizing the system and keeping the performance at an acceptable level. Because of this, fault-tolerant control methods have been an active area during the last decade, and several significant contributions to the reliability of safety-critical systems have been made.

Model predictive control (MPC) has shown to be a powerful control scheme for multi-variable control problems, and provide a natural framework for integrating receding-horizon optimization, while also achieving system reconfiguration in the event of faults. However, almost all the efforts on incorporating fault-tolerance in MPC are focused on reactive fault-tolerance, which aim to handle a fault *after* it has occurred. In contrast, proactive fault-tolerant control seeks to utilize an estimated, conservative time window between the warning of an incipient fault and the time at which the faulty component is rendered useless to steer the state inside a recoverable region *before* the fault occurs. As such, a proactive approach circumvents the issues of possible infeasibilities and destabilization often encountered in reactive approaches, while allowing the system to continue operation during the subsequent system repair.

Furthermore, economic MPC (EMPC) has received increasing attention in the recent years. Rather than separating real-time optimization and control, an economic MPC scheme merges dynamic economic operations with the feedback properties of conventional MPC. However, there has not been paid much attention to including fault-tolerance and economic optimization in a unified framework.

This thesis proposes a proactive EMPC algorithm for handling incipient faults, that also takes economic profits in to account. The scheme utilizes an exact penalty function to steer the system inside an invariant set ensuring stability during the loss of actuation in the system. Additionally, the scheme is extended to be robust in terms of handling unknown disturbances to the system, while still achieving the desired fault-tolerance. Stability for the proposed scheme is proven, both for systems with and without disturbances. The merits and shortcomings of the proposed scheme is demonstrated through several numerical examples.

Sammendrag

I et system hvor ett eller flere pådragsorgan feiler, er vanlige kontrollalgoritmer ikke lenger tilstrekkelige for stabilisering av systemet. Dette har ført til økende forskning på feiltolerante kontrollsystemer i de siste ti årene.

Modell prediktiv regulering (MPC) har vist seg å være en fleksibel regulator for multi-variable systemer, og gir et godt rammeverk for å integrere optimalisering av systemdynamikken med feiltolerant kontroll. Hittil har nesten all forskning på feiltolerant MPC vært på såkalte reaktive metoder, hvor målet er å håndtere en feil etter den har forekommet. I kontrast til dette er målet med proaktive feiltolerante metoder å utnytte tidsintervallet mellom en advarsel om en påbegynnende feil, og tidspunktet da feilen faktisk skjer. Regulatoren vil i denne tidsperioden styre systemet inn i et område hvor de resterende pådragene er i stand til å stabilisere systemet når feilen oppstår. På denne måten unngår proaktive metoder potensielle ustabile systemresponser som ofte oppstår med reaktive metoder, i tillegg til å tillate kontinuerlig systemoperajon under påfølgende systemraperasjon.

Videre har økonomisk MPC (EMPC) fått økende oppmerksomhet de siste årene. Isteden for å separere sanntidsoptimalisering og automatisk styring, er målet med EMPC å slå disse sammen i et felles rammeverk. Det har generelt blitt utført lite forskning på å inkludere feiltoleranse i et slikt rammeverk.

Denne oppgaven presenterer en proactiv EMPC kontrollalgoritme for å håndtere påbegynnende feil i systemer, samtidig som den tar den økonomiske gevinsten fra systemet i betraktning. Metoden benytter en eksakt straffefunksjon til å styre systemet inn i et invariant område, der stabilitet under pådragsfeil er garantert. Videre er algoritmen utvidet til å være robust i form av å håndtere forstyrrelser i systemet, og samtidig oppnå ønsket feiltoleranse. Rapporten inneholder stabilitetsbevis for metoden, både for systemer med og uten forstyrrelser. Teorien er illustrert med nummeriske eksempler, og styrker og svakheter med metoden er diskutert.

Contents

Preface i
Abstract ii
Sammendrag iv
Abbreviations x

1 Introduction 1
1.1 Faults and failures 1
1.2 Model predictive control 2
1.3 Economic model predictive control 3
1.4 Fault-tolerant control 4
1.4.1 Fault modeling 5
1.4.2 Fault-tolerant MPC 5
1.5 Introducing the illustrative examples 8
1.5.1 System with two states, two inputs and one input dropout . . 8
1.5.2 System with two states, three inputs and two input dropouts 9
1.6 Report outline 11

2 Invariant Set Theory and MPC Feasibility 13
2.1 Introduction 13
2.2 Feasibility in MPC 15
2.3 Defining the safety set 16
2.4 Computing control invariant sets 17
2.5 Numerical illustrative examples 20
2.5.1 Example with two states, two inputs and one dropout 20
2.5.2 Example with two states, three inputs and two dropouts . . . 20

3 Soft Constraints and Penalty Functions 23
3.1 Introduction 23
3.2 Exact penalty functions 24
3.3 Soft constrains and penalty functions in MPC 25
3.4 Computing a lower bound for the penalty weight 28
3.4.1 Preliminaries 29
3.4.2 Using bi-level and mixed integer programming 32
3.4.3 Adding an extra minimization problem 33
3.4.4 Adding an explicit LICQ constraint 34

3.5 Numerical illustrative examples . . . 36
3.5.1 Example with two states, two inputs and one dropout . . . 36
3.5.2 Example with two states, three inputs and two dropouts . . . 36

4 Proactive Fault-tolerant Economic MPC 39
4.1 Introduction . . . 39
4.2 Approach . . . 41
4.2.1 Nominal operation . . . 41
4.2.2 Safe operation . . . 42
4.2.3 Fault operation . . . 44
4.3 Multiple actuator faults . . . 46
4.4 Stability . . . 48
4.5 Numerical illustrative examples . . . 51
4.5.1 Example with two states, two inputs and one dropout . . . 51
4.5.2 Example with two states, three inputs and two dropouts . . . 54
4.5.3 Example with three states, three inputs and one dropout . . . 57

5 Robust Proactive Fault-tolerant Economic MPC 61
5.1 Brief review of disturbance-handling in MPC . . . 61
5.2 Tube-based robust economic MPC . . . 62
5.3 Approach . . . 66
5.3.1 Fault-free operation . . . 66
5.3.2 Safe operation . . . 67
5.3.3 Fault operation . . . 68
5.4 Stability . . . 69
5.5 Numerical illustrative example . . . 70

6 Discussion 77
6.1 On the assumptions for the approach . . . 77
6.2 Implementing the scheme and numerical results . . . 79
6.2.1 Without disturbances . . . 79
6.2.2 With disturbances . . . 80

7 Conclusion 83

8 Future work 85

Bibliography 86

A MPC formulation 93
A.1 Compact quadratic cost formulation . . . 93
A.2 Compact linear cost formulation . . . 94

B KKT-conditions 97

C Set operations 99

D Discrete minimal-time control 101

E Conference paper 103

Abbreviations

FTC	=	Fault-tolerant control
FDI	=	Fault detection and isolation
MPC	=	Model predictive control
EMPC	=	Economic model predictive control
FTMPC	=	Fault-tolerant model predictive control
FTEMPC	=	Fault-tolerant economic model predictive control
RTO	=	Real-time optimization
RPI	=	Robust positively invariant
mRPI	=	Minimal robust control invariant
RCI	=	Robust control invariant
LP	=	Linear program
MILP	=	Mixed integer linear program
DMTC	=	Discrete minimal-time control

Chapter 1

Introduction

In safety-critical systems, it is crucial that some level of performance is maintained in the event of faults. Systems such as chemical plants and nuclear power plants need to have fault-handling as the top priority. Malfunctions in actuators greatly reduce safety, and the economic losses can be severe. Fault-tolerant control (FTC) schemes for handling these types of critical situations are therefore needed to avoid potential catastrophic events.

This chapter gives a short introduction to the concepts, methodologies and notations used in this thesis, and provides the motivation for the proposed theory. In particular, fault-tolerant model predictive control (FTMPC) is introduced, which is a way of incorporating fault-tolerance in model predictive control (MPC). The main contributions of this thesis are introduced, which will be built on throughout the report. Since the thesis is a continuation of an earlier project by the same author, the explanations in this chapter will be rather brief, and the reader is referred to the earlier project thesis contained in the digital attachments, for more detailed descriptions. First, some basic definitions of faults and failures are provided.

1.1 Faults and failures

Isermann and Ballé (1997) define fault and failure, in compliance with the definitions given by the IFAC SAFEPROCESS technical committee, in the following way:

Fault:

An unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/standard condition.

Failure:

A permanent interruption of a system's ability to perform a required function under specified operating conditions.

It is clear from the definition that a failure is a condition that is much more severe than a fault. When a fault occurs in a component for example, the component may still be usable, but becomes less effective. For a failure however, a totally different component is needed to achieve the same objective. Failure might also lead to the need for system shut-down. Throughout this report, a "fault" will be used when something is wrong with a component, and "failure" will typically be described as the system becoming unstable due to a fault not being compensated for by the controller.

In addition, it is important to make the following distinction between an incipient and an actual fault:

Incipient fault:
A fault that is about to occur. For example because of wear and tear on components, or an actuator that is about to be taken out of action for maintenance.

Actual fault:
A fault that *has* occurred in the system, a sensor, or an actuator. This will often times severely reduce the effectiveness of a controller, and may in the worst cases lead to system failure.

Different types of actions need to be taken to avoid failure from actual and incipient faults. The handling of actual faults has to rely on fast reconfiguration or robustness of the controller. For incipient faults, however, it may sometimes be possible to handle the fault or at least prepare the system for the upcoming fault before it occurs. This will be introduced in Section 1.4.2 and is the main contribution of this work.

1.2 Model predictive control

Model Predictive control is one of the most commonly used control algorithms for multi-variable control problems. Mellichamp et al. (2010) summarizes the MPC concept as follows: A multiple input, multiple output system is to be controlled while satisfying constraints on the input and output variables. The algorithm uses a model of the system to predict the future states. At each timestep, an optimization problem is solved that takes future events into account and calculates a sequence of control moves to take the system from the current state to the reference state while minimizing some cost function. The first control move is then applied to the system, and the whole problem is recomputed in the next timestep. The main reason for the popularity of the MPC controller is its effectiveness in handling large multi-variable systems with constraints on both inputs and states.

This section briefly describes the notations used for MPC in this thesis. It is assumed that the reader is familiar with the general MPC principle and no detailed explanation is therefore given. For a more detailed description, the reader is referred to Maciejowski (2002).

Consider the discrete linear system

$$x_{k+1} = Ax_k + Bu_k, \tag{1.1}$$

with $k \in \{0, 1, \dots\}$, $x_k \in \mathbb{R}^n$ is the system state, $u_k \in \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times n}$ are constant matrices. The system is subject to linear inequality constraints on the states and inputs:

$$x \in \mathbb{X} \triangleq \{x \in \mathbb{R}^n \mid Dx \leq d\} \tag{1.2a}$$

$$u \in \mathbb{U} \triangleq \{u \in \mathbb{R}^m \mid Hu \leq h\}, \tag{1.2b}$$

where $d \in \mathbb{R}^{n_d}$ and $h \in \mathbb{R}^{n_h}$ define the constraints, with n_d and n_g denoting the number of state and input constraints, respectively. The matrices $D \in \mathbb{R}^{n_d \times n}$ and $H \in \mathbb{R}^{n_h \times m}$ are the state and input constraint distribution matrices. At each timestep, the following optimization problem is solved:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) & & \text{(1.3a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, & \forall k \in \{0, \dots, N-1\} & \quad \text{(1.3b)}\\
& x_0 = x_{\text{init}}, & & \text{(1.3c)}\\
& x_k \in \mathbb{X}, & \forall k \in \{1, \dots, N-1\} & \quad \text{(1.3d)}\\
& u_k \in \mathbb{U}, & \forall k \in \{0, \dots, N-1\} & \quad \text{(1.3e)}\\
& x_N \in \mathcal{X}_{\text{terminal}}, & & \text{(1.3f)}
\end{aligned}$$

where $\mathbf{x} = \begin{bmatrix} x_1 & x_1 & \dots & x_N \end{bmatrix}^T$, $\mathbf{u} = \begin{bmatrix} u_0 & u_1 & \dots & u_{N-1} \end{bmatrix}^T$, and $(\mathbf{x}^*, \mathbf{u}^*)$ denotes the optimal solution to (1.3). Furthermore, $l(x_k, u_k)$ is a stage cost function, typically defined as the quadratic difference between the predicted state of the system and a reference state, or describing the economic profits/losses of the system. The vector x_{init} is the current state of the system and N is the horizon. Equal prediction and control horizon is assumed. Additionally, $x_N \in \mathcal{X}_{\text{terminal}}$ defines a terminal constraint for the problem, and is included for stability purposes, see Mayne et al. (2000). This will be further explained in the case of an economic cost function in Chapter 4.

At each timestep the input

$$u_e(x_{\text{init}}) = u_0^* \tag{1.4}$$

is applied to the system. The finite-horizon problem (1.3) is then repeatedly re-optimized in a receding-horizon manner with current state (1.3c) updated through measurements of x. *State feedback is assumed in the rest of this thesis.*

1.3 Economic model predictive control

MPC with a quadratic stabilizing/tracking cost function has been widely used over the last decades. For a tracking MPC, the optimal reference state is typically generated by an external real-time optimization layer (RTO) and sent to be tracked by the MPC. While the standard tracking MPC allows for tunable closed-loop response, it may not be an adequate representation of managing real-time process operation with respect to the process economic performance (Ellis and Christofides, 2014). A positive deviation from the reference may represent a profit, while a negative deviation from the reference may represent a loss (Siirola and Edgar, 2012). Recently there has been increased attention to economic model predictive control (EMPC), which contrary to separated RTO and MPC, merges dynamic economic operations with the feedback properties of conventional MPC and incorporates an economic cost in its formulation (Mayne, 2014; Amrit et al., 2011; Angeli et al., 2012; Diehl et al., 2011).

However, unlike standard tracking MPC, stability results for EMPC is still a researched topic. In standard MPC, the stability proof relies on the cost function being positive definite and the cost function can therefore be used as a Lyapunov function for the system. This is not the case in EMPC, and the same proof can not be used. Nominal stability of EMPC has been proved for systems with a terminal equality constraint, satisfying strong duality (Diehl et al., 2011) or strict dissipativity (Angeli et al., 2012), or with a terminal cost and inequality constraints for systems satisfying strict dissipativity (Amrit et al., 2011). Chapter 4 provides more stability theory for EMPC, and in particular on how to use this theory in a fault-tolerant context.

Furthermore, most research on EMPC has been aimed at nominal systems without faults and disturbances. To the authors knowledge, few publications exist on how to incorporate fault-tolerance and disturbance handling in EMPC. It has, however, recently began to receive increasing attention. This thesis illustrates a scheme for implementing fault-tolerance in an EMPC framework, which is also extended to incorporate disturbance attenuation based on a recently published paper (Bayer et al., 2014).

1.4 Fault-tolerant control

As earlier described, faults can cause severe failures in a system. A conventional feedback control design might result in poor performance, or even instability for a system that is affected by faults. A structured and robust fault-tolerant detection and control scheme enhances reliability and continuity of system operations, both for safety-critical processes and for chemical production and manufacturing (Zhang and Jiang, 2008). This has motivated a considerable amount of research in the last decade on how to incorporate fault-tolerance in controllers, and thereby have the system retain an acceptable level of performance in the case of faults. Zhang and Jiang (2008) define a fault-tolerant control system as a closed-loop control system that can tolerate component malfunctions, while maintaining desirable performance and stability properties. The design of these controllers are critical for reliable operation of many systems. This section briefly introduces how to model faults using mathematical equations, and describe how fault-tolerance can be included in an MPC framework for handling both actual and incipient actuator faults.

1.4.1 Fault modeling

The focus of this thesis is to handle actuator faults, which need to be modeled in an effective manner. The following notations for changes in system dynamics and constraints when a fault occurs are used. Given a fault in actuator $j \in \{1, 2, 3, \ldots m\}$, the system is modeled as

$$x_{k+1} = Ax_k + B_j u_k, \tag{1.5}$$

where $B_j \in \mathbb{R}^{m \times n}$ is a constant matrix describing how the system reacts to an input when a fault in actuator j is present. The modeling of the constraints is given by a change in the constraint set $\mathbb{U}$, where $\mathbb{U}_j$ denotes the new constraints after the impact of the fault. In the case of multiple faults, these are included in addition to the original fault, e.g. B_{ji} denotes a fault in actuator j and i.

1.4.2 Fault-tolerant MPC

Due to the receding-horizon nature of MPC, it allows for effective ways of incorporating fault-tolerance in the controller (Maciejowski, 1999). Its ability to efficiently handle complex systems with hard control constraints and many inputs and outputs allows for direct on-line adaptation of the controller to faults in the system. Faults are represented as constraints and/or a changes in the internal model in the optimization problem, as described in the previous section. The MPC then effectively computes a new control law to accommodate the fault. There are several examples in the literature on the use of MPC in this context, including Miksch et al. (2008).

Furthermore, active fault-tolerant control methods can broadly be classified as (Lao et al., 2013):

- Reactive
- Proactive.

Reactive approaches try to minimize the impact of a fault after it occurs, relying on a fault detection and isolation unit (FDI) and reconfiguration of the control system. Proactive fault-tolerant control methods employ an FDI unit to detect slowly developing, degradation of performance in process components, actuators or sensors that indicates an incipient fault (Demetriou and Polycarpou, 1998), together with a probabilistic prediction method for the time of the incipient fault, e.g., Salfner and Malek (2007). Contrary to reactive methods, a proactive control scheme takes proactive action to prevent negative impact of the predicted fault situation. Proactive fault-tolerant methods is emerging as a complement to reactive schemes for designing robust and effective fault control control. However, it is important to emphasize that proactive approaches is intended only to supplement reactive schemes capable of handling abrupt faults. A proactive method can, on the other hand, be efficiently applied for maintaining process operation, minimize down time or prevent shut-downs in terms of certain types of faults, and also to perform scheduled maintenance. The next sections will elaborate on how to use MPC for reactive and proactive FTC.

Reactive FTMPC

As earlier described, MPC allows for efficient reconfiguration to handle faults, and in particular, actuator faults. Once a fault is detected and its magnitude is estimated, the MPC is reconfigured on-line to redistribute the task of the faulty component to the remaining healthy ones. Given a detected actual fault in actuator j, mathematically modeled as described in Section 1.4.1, the following measure is taken in the MPC problem to compensate the fault. Recall the nominal MPC formulation (1.3) for the fault-free system. By modifying the constraints as in Section 1.4.1, the problem to be solved at each timestep becomes:

$$\begin{aligned}
&\min_{\mathbf{x},\mathbf{u}} && \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(1.6a)}\\
&\text{s.t.} && x_{k+1} = Ax_k + B_j u_k, && \forall k \in \{0, \dots, N-1\} && \text{(1.6b)}\\
& && x_0 = x_{\text{init}}, && && \text{(1.6c)}\\
& && x_k \in \mathbb{X}, && \forall k \in \{1, \dots, N-1\} && \text{(1.6d)}\\
& && u_k \in \mathbb{U}_j, && \forall k \in \{0, \dots, N-1\} && \text{(1.6e)}\\
& && x_N \in \mathcal{X}^j_{\text{terminal}} && && \text{(1.6f)}
\end{aligned}$$

which is then is then repeatedly re-optimized in a receding-horizon manner until the fault is fixed, or an additional fault occurs. The term, $x_N \in \mathcal{X}^j_{\text{terminal}}$, is as described in Section 1.2 a terminal constraint added for stability purposes.

Remark 1.1. Of course, this assumes that the remaining actuators are able to control the faulty system when the fault occurs. This might not always be the case, and the next section introduces proactive fault-tolerant control. This approach aims to prepare the system for the fault before it occurs so that the remaining actuators are in fact able to control the faulty system.

Proactive FTMPC

An actuator fault may cause loss of controllability at the current operating point and thereby destabilize the system. This is due to the fact that when loss of actuators occur, the region in which the system can be controlled will often be reduced because the remaining actuators are not powerful enough to control the system due to the nominal constraints on the inputs. This is a serious issue, especially for open-loop unstable systems, which are dependent on effective control actions in order to be kept stable. In such situations, a plant emergency mode or shut-down is necessary in order to avoid component damage and safety threatening situations.

An alternative solution to this potential issue would be to operate the system within a set that guarantees controllability for the different kinds of actuator faults. This approach, however, would generally be conservative and detrimental for the nominal economic operation of a system. If information about an upcoming fault is available, another alternative is to employ a proactive scheme for actuator faults (Lao et al., 2013), which is obtained by allowing the system to operate outside the guaranteed stability region with one of the actuators inactive, but force the system inside this region upon indication of an incipient fault. The proactive FTMPC in Lao et al. (2013) uses Lypanunov-based MPC with predesigned controllers to drive the system inside this safety system, while Bø and Johansen (2014) develops a hybrid scheme with scenario based safety constraints and reconfigurable control.

The focus of this thesis is to formulate and analyze a proactive fault-tolerant economic MPC (FTEMPC) scheme based on *soft constraints* and *penalty functions* to obtain an efficient way of incorporating proactive fault-tolerance in an EMPC framework. The basic idea is to separate the operation modes of the controller in to the following categories:

- *Nominal operation*: When no fault or warning about an incipient fault is present, the system will operate in nominal operation. The MPC will drive the system to an optimal steady-state subject to a fault-free system and nominal constraints.

- *Safe operation*: When a warning about a known incipient fault is received, the system is driven into a *safety set* using soft constraints and a penalty function, in which the remaining actuators can stabilize the system when the fault occurs.

- *Fault operation*: When the fault *actually* occurs, the MPC formulation is updated to include the constraints which are introduced by the fault, in the same manner as for reactive fault-tolerant MPC described previously.

A rule for switching between MPC problems for nominal, safe-mode transition and faulty operations is designed, based on the assumption of a separate available FDI unit to indicate and distinguish incipient and actual faults. Criteria for how to make sure that the system is driven into the safety set, and also how to define and compute this safety set is analyzed. The approach relies on theory from the following fields:

- *Exact penalty functions* (Pietrzykowski, 1969), used to compute a lower bound on the penalty parameter in order to make sure that the system is steered into the safety set upon warning of an incipient fault;

- *Set theory*, and in particular, *invariant sets* (Kerrigan and Maciejowski, 2000a) used for defining the safety set.

This theory will be provided in the following chapters before the main approach is designed and analyzed both for systems with and without disturbances.

1.5 Introducing the illustrative examples

In order to illustrate the theoretic concepts, two recurring illustrative examples are used throughout the thesis, where the different parts of the implementation are described at the end of its respective chapter. The following cases will be investigated:

- System with two states, two inputs and one input dropout;
- System with two states, three inputs and two input dropouts.

The end goal for each example is to have a working proactive fault-tolerant scheme to handle incipient faults. Thus the safety sets as well as penalty functions need to be designed. As can be seen, the focus has been put on systems with two states. This is due to the fact that they are easily visualized in the plane, and provide a good framework for illustration and discussion. The examples include only open-loop unstable systems, i.e. linear system matrices with eigenvalues outside the unit circle. The reason for this is that the regions in which a controller can stabilize the system once a fault occurs change considerably more for these systems than for systems that are open-loop stable. Thus, the examples will give a better illustration for unstable systems.

Additionally, in order to illustrate how the approach scales for larger systems, a system with three states is also studied in Chapter 4. The three aforementioned examples illustrate the approach for systems without disturbances. Additional theory is needed to implement the approach for systems with disturbances, an example illustrating the approach for this case is included in Chapter 5 after the required theory is presented.

1.5.1 System with two states, two inputs and one input dropout

This example illustrates the scheme for a system with two states, two inputs and a single actuator dropout.

System description

Consider the linear time-invariant discrete system

$$x_{k+1} = Ax_k + Bu_k, \tag{1.7}$$

where $x \in \mathbb{R}^2$ is the system state and $u \in \mathbb{R}^2$ is the system input. A and B are constant matrices given as

$$A = \begin{bmatrix} 1.3337 & 0.9443 \\ 0.5902 & 1.3337 \end{bmatrix} \tag{1.8}$$

$$B = \begin{bmatrix} -0.2572 & -0.3817 \\ -0.2665 & -0.1954 \end{bmatrix}. \tag{1.9}$$

The eigenvalues of A are 2.0802 and 0.5872, and so the system is open-loop unstable. The constraints on x and u are

$$x \in \mathbb{X} = \left\{ x \mid \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 6 \\ 6 \end{bmatrix} \right\} \tag{1.10}$$

$$u \in \mathbb{U} = \left\{ u \mid \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 15 \end{bmatrix} \right\} \tag{1.11}$$

The objective is to design a proactive fault-tolerant economic MPC that is able to accommodate incipient and actual faults by driving the system into a safety set before the fault occurs while still optimizing process economics.

Economic cost function

The following economic cost function is used in the MPC formulation

$$J = \sum_{k=0}^{N-1} l\left(x_k, u_k\right) = \sum_{k=0}^{N-1} \left(-q x_k + \left\|R u_k\right\|_1\right), \tag{1.12}$$

where $N = 10$, $q = \begin{bmatrix} 10 & 10 \end{bmatrix}$, $R = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$. Note that this is a linear, not positive definite function, as is the case for most economic cost functions.

Fault modeling

Consider a situation where the second actuator u_2 has a dropout at time t_f, the warning about the incipient fault is received at t'. Due to the complete dropout of the second actuator, the input constraints after the fault has occurred are given by

$$u \in \mathbb{U}_2 = \left\{ u \mid \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 0 \end{bmatrix} \right\} \tag{1.13}$$

and the input matrix becomes

$$B_2 = \begin{bmatrix} -0.2572 & 0 \\ -0.2665 & 0 \end{bmatrix}. \tag{1.14}$$

As described in Section 1.4.2, both the safety set subject to a dropout of the second actuator, as well as a penalty function that guarantees that the system will be steered into the safety set, need to be defined. This is done in Chapters 2 and 3 before the complete implementation of the control scheme is implemented in Chapter 4.

1.5.2 System with two states, three inputs and two input dropouts

This example illustrates the scheme for a system with two states, three inputs and two actuator dropouts.

System description

Consider the linear time-invariant discrete system

$$x_{k+1} = Ax_k + Bu_k, \tag{1.15}$$

where $x \in \mathbb{R}^2$ is the system state, $u \in \mathbb{R}^2$ is the system input. A and B are constant matrices given as

$$A = \begin{bmatrix} 1.1494 & 0.3349 \\ 0.4465 & 1.1717 \end{bmatrix} \tag{1.16}$$

$$B = \begin{bmatrix} -0.0660 & -0.4636 & -0.2286 \\ -0.0966 & -1.0303 & -0.1502 \end{bmatrix}. \tag{1.17}$$

The eigenvalues of A are 0.7737 and 1.5474, and so the system is open-loop unstable. The constraints on x and u are

$$x \in \mathbb{X} = \left\{ x \mid \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 15 \\ 15 \end{bmatrix} \right\} \tag{1.18}$$

$$u \in \mathbb{U} = \left\{ u \mid \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \leq \begin{bmatrix} 25 \\ 10 \\ 5 \end{bmatrix} \right\} \tag{1.19}$$

Economic cost function

The following economic cost function is used in the MPC formulation

$$J = \sum_{k=0}^{N-1} l\left(x_k, u_k\right) = \sum_{k=0}^{N-1} \left(-qx_k + \|Ru_k\|_1\right), \tag{1.20}$$

where $N = 10$, $q = \begin{bmatrix} 10 & 10 \end{bmatrix}$, $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

Fault modeling

The situation considered is a dropout of actuator 2, followed by an additional dropout of actuator 1. The constraint on the input introduced by the first fault is given by

$$u \in \mathbb{U}_2 = \left\{ u \mid \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \leq \begin{bmatrix} 25 \\ 0 \\ 5 \end{bmatrix} \right\} \tag{1.21}$$

and the input matrix becomes

$$B_2 = \begin{bmatrix} -0.0660 & 0 & -0.2286 \\ -0.0966 & 0 & -0.1502 \end{bmatrix}. \tag{1.22}$$

Additionally, when actuator 1 has a dropout, the constraints on the inputs become

$$u \in \mathbb{U}_{12} = \left\{ u \mid \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} \right\} \tag{1.23}$$

And the input matrix

$$B_{12} = \begin{bmatrix} 0 & 0 & -0.2286 \\ 0 & 0 & -0.1502 \end{bmatrix}. \tag{1.24}$$

As for the previous example, the safety set and the penalty function are computed in Chapters 2 and 3, respectively, before the complete implementation of the control scheme is implemented in Chapter 4.

1.6 Report outline

The report is organised as follows: Chapter 2 defines various types of invariant sets and their importance in MPC design, and in particular for defining the safety set in the proactive FTEMPC scheme. The chapter also presents algorithms for computing these sets, and designs the safety set for the two recurring examples. Chapter 3 gives an introduction and describes the theory of soft constraints and penalty functions in MPC. A method for computing a lower bound on the penalty parameter is described and analyzed, and applied to the recurring examples. Furthermore, the proactive FTEMPC approach is described in detail in Chapter 4 for disturbance-free systems, where the recurring examples are fully implemented as well as an additional example. The scheme is extended to systems with bounded disturbances in Chapter 5, and includes an illustrative example. Finally, a discussion of the results and the assumptions that were made are included in Chapter 6, and Chapter 7 concludes the report and outlines potential future work.

Chapter 2

Invariant Set Theory and MPC Feasibility

Section 1.4.2 briefly introduced proactive FTEMPC, where the objective is to take proactive action and thereby guarantee stability of the system when an actuator fault occurs. The safety set was defined to be a set where the remaining actuators are able to control the system once one, or possibly multiple actuators have dropouts. This chapter is focused on *positively invariant set* theory, and the feasibility of the MPC problem. It will be clear that positively invariant sets play an important part when it comes to MPC feasibility and for designing the safety set. First, the definitions for the most important types of positively invariant sets used in this thesis are given, followed by criteria for feasibility in MPC and how to choose the safety set. The last section describes general algorithms for computing these sets.

2.1 Introduction

The properties of positively invariant sets are involved in many different problems in control theory. Given a dynamic system, a subset of the state space is said to be positively invariant if it has the property that, if it contains the system state at some time, then it will contain it also in the future (Blanchini, 1999). This provides important insights to system behavior, especially for constrained systems. In general, given a constrained dynamic system, not all trajectories originating from an initial state that satisfies the constraints will continue to satisfy these constraints in the future. However, if the initial state satisfies its constraints *and* lies in a positively invariant set, one can then guarantee that all trajectories originating from this initial condition will satisfy the constraints for all time. Definition 2.1 defines a positively invariant set. Let Ω and Ξ denote any arbitrary subsets of $\mathbb{R}^n$, and $\mathbb{N}_+$ the set of positive integers.

Definition 2.1 (Positively invariant set (Blanchini, 1999))**.** The non-empty set

$\Omega \subset \mathbb{R}^n$ is positively invariant for the autonomous system $x_{k+1} = f(x_k)$ if and only if $\forall x_0 \in \Omega$, the system evolution satisfies $x_k \in \Omega, \forall k \in \mathbb{N}_+$.

Furthermore, many systems have clearly defined hard constraints on both the states and inputs. The objective in constrained control is to design controllers such that for a given initial condition, the controller is able to control the system while satisfying the constraints. In these types of problems, one often considers *control invariant sets* in order to effectively design control laws. A subset of the state space is said to be control invariant if there exists a control law that will keep all trajectories originating from the set inside the same set while satisfying the constraints. By determining the aforementioned set, one can define operating regions where the constraints are sure to be satisfied.

Definition 2.2 (Control invariant set (Blanchini, 1999))**.** The non-empty set $\Omega \subset \mathbb{R}^n$ is a control invariant set for the system $x_{k+1} = f(x_k, u_k)$ if and only if there exists a feedback control law $u_k = g(x_k)$ such that Ω is a positively invariant set for the closed-loop system $x_{k+1} = f(x_k, g(x_k))$ and $u_k \in \mathbb{U}, \forall x_k \in \Omega$.

Additionally, one is often interested in finding the largest control invariant set containing all control invariant sets, this set is of great importance when designing operating regions for a system, and in particular, in MPC. This leads to the definition of the maximal control invariant set.

Definition 2.3 (Maximal control invariant set (Blanchini, 1994))**.** The non-empty set $\mathbb{C}_\infty(\Omega)$ is the maximal control invariant set contained in Ω for the system $x_{k+1} = f(x_k, u_k)$ if and only if $\mathbb{C}_\infty(\Omega)$ is control invariant and contains all control invariant sets contained in Ω, i.e. Ξ is control invariant only if $\Xi \subseteq \mathbb{C}_\infty(\Omega) \subseteq \Omega$

In MPC, control invariant set theory is critical for ensuring feasibility of the MPC problem and has been very successful in providing sufficient nominal and robust feasibility and stability conditions (Mayne et al., 2000). This is due to the fact that state and control constraints can be satisfied if and only if the initial state belongs to a control invariant set for the system Kerrigan and Maciejowski (2000a). Furthermore, in the case of actuator faults, one can think of the current state of the system when the fault occurs as the initial state for the new, faulty system. It is therefore critical that the system operates in a control invariant set subject to the constraints introduced by the fault, when the fault occurs. This will be important for determining the safety set.

For systems that are affected by disturbances, the aforementioned definitions are adapted in the following way. Let $w_k \in \mathbb{W} \in \mathbb{R}^p$ denote the disturbance at time k, where $\mathbb{W}$ denotes the set of possible values for the disturbances.

Definition 2.4 (Robust positively invariant set, (Kerrigan and Maciejowski, 2000a))**.** The non-empty set $\Omega \subset \mathbb{R}^n$ is robust positively invariant for the system $x_{k+1} =$

$f(x_k, w_k)$ if and only if $\forall x_0 \in \Omega$ and $\forall w_k \in \mathbb{W}$, the system evolution satisfies $x_k \in \Omega, \forall k \in \mathbb{N}_+$.

Definition 2.5 (Robust control invariant set (Kerrigan and Maciejowski, 2000a))**.** The set $\Omega \subset \mathbb{R}^n$ is a robust control invariant set for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if there exists a feedback control law $u_k = h(x_k)$ such that Ω is a robust positively invariant set for the closed-loop system $x_{k+1} = f(x_k, h(x_k), w_k)$ and $u_k \in \mathbb{U}, \forall x_k \in \Omega$.

Definition 2.6 (Maximal robust control invariant set (Kerrigan and Maciejowski, 2000a))**.** The set $\tilde{\mathbb{C}}_\infty(\Omega)$ is the maximal robust control invariant set contained in Ω for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if $\tilde{\mathbb{C}}_\infty(\Omega)$ is robust control invariant and contains all the robust control invariant sets contained in Ω.

Additionally, the use of positively invariant sets for control design is dependent on the existence of algorithms for computing these sets. There exist several contributions to this field, including Kerrigan and Maciejowski (2000a); Athanasopoulos and Bitsoris (2010); Scibilia et al. (2011). A general framework for these computations are given in Section 2.4.

2.2 Feasibility in MPC

The definitions in the previous section are key ingredients for analyzing the feasibility and stability of MPC. This section presents some general well-known results for nominal MPC feasibility. Consider again the MPC problem defined in Chapter 1, repeated here for convenience:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && (2.1a)\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0, \dots, N-1\} && (2.1b)\\
& x_0 = x_{\text{init}}, && && (2.1c)\\
& x_k \in \mathbb{X}, && \forall k \in \{1, \dots, N-1\} && (2.1d)\\
& u_k \in \mathbb{U}, && \forall k \in \{0, \dots, N-1\} && (2.1e)\\
& x_N \in \mathcal{X}_{\text{terminal}}. && && (2.1f)
\end{aligned}$$

Often one is interested in obtaining the set of states for which the MPC problem is feasible, as is the case when defining the safety set. The *admissible* set, $\mathbb{Z}_N$, of the MPC problem is the set of states and inputs that satisfies all constraints, i.e.

$$\mathbb{Z}_N = \{x_0, \mathbf{u} \mid \exists \mathbf{x} \text{ such that } (\mathbf{x}, \mathbf{u}) \text{ satisfies (2.1b) – (2.1f)}\} \tag{2.2}$$

The set of feasible states, denoted the feasible set $\mathbb{X}_N$, can therefore be interpreted as the orthogonal projection of $\mathbb{Z}_N$ onto $\mathbb{R}^n$ (Kerrigan and Maciejowski, 2000a)

$$\mathbb{X}_N = \{x_0 \mid \text{such that } (\mathbf{x}, \mathbf{u}) \in \mathbb{Z}_N\}, \tag{2.3}$$

where the subscript is included to stress the dependence on N.

Remark 2.1. Note the difference between $\mathbb{X}$ and $\mathbb{X}_N$. The former denotes the allowable states without considering the other constraints. The latter denotes the allowable states where the rest of the constraints are also satisfied.

As mentioned in Section 2.1, the MPC problem is feasible if and only if the initial state belongs to a control invariant set for the system. From Definition 2.3 it is therefore clear that the initial state needs to lie in the maximal control invariant set. Hence, the problem is feasible if and only if

$$x_0 \in \mathcal{C}_\infty(\mathbb{X}). \tag{2.4}$$

However, due to the finite-horizon nature of MPC, the control at the next time instant could be different from the previously computed value, even without disturbances (Kerrigan and Maciejowski, 2000a). This can result in a situation where $x_1 \in \mathbb{X} \setminus \mathbb{X}_N$, which will result in an infeasible problem at the next time instant. Additionally, it is possible that the feasible set is not a subset of the maximum control invariant set, which will result in a trajectory that does not stay in $C_\infty(\mathbb{X})$, i.e. $x_1 \in \mathbb{X}_N \nsubseteq C_\infty(\mathbb{X})$, and the MPC problem will become infeasible at the next timestep.

It is critical to design the MPC such that it is feasible for all time in order to avoid situations where the MPC is no longer able to find an admissible control input. A well known result that guarantees this to hold is by choosing a control invariant terminal set $x_N \in \mathcal{X}_{\text{terminal}}$ (Mayne et al., 2000). The feasible set will then be control invariant and can be interpreted as the maximal controlled invariant set by means of the MPC with prediction horizon N and terminal set $\mathcal{X}_{\text{terminal}}$ (Scibilia et al., 2011).

Thus by choosing an initial condition for the system that lies in the maximum control invariant set, as well as by choosing a control invariant terminal set, the MPC will be feasible for all time and thereby guarantee stability. Note that the same result may be achieved by using a terminal cost or a long prediction horizon Mayne et al. (2000). However, this thesis will focus on the use of a terminal constraint. This will be clear in Chapter 4, where a terminal constraint is used to guarantee stability for economic MPC.

2.3 Defining the safety set

As described in the previous section, positively invariant sets play an important part in the theory of feasibility for the MPC problem. It was shown that the MPC problem is feasible if the initial condition lies in a control invariant set for the system, and that it is feasible for all time if it is initially feasible and the terminal constraint is chosen to be a control invariant set. This section describes the choice of the safety set $\mathbb{S}_j$ based on the aforementioned invariant set theory.

In Section 1.4.2, a loss of actuators was described to change the operating conditions of the system, and thus also often the set in which the MPC problem is feasible. Recall the reactive reconfiguration of the MPC in order to compensate for a fault in actuator j, as introduced in Chapter 1:

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(2.5a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + B_j u_k, && \forall k \in \{0, \dots, N-1\} && \text{(2.5b)}\\
& x_0 = x_{\text{init}}, && && \text{(2.5c)}\\
& x_k \in \mathbb{X}, && \forall k \in \{1, \dots, N-1\} && \text{(2.5d)}\\
& u_k \in \mathbb{U}_j, && \forall k \in \{0, \dots, N-1\} && \text{(2.5e)}\\
& x_N \in \mathcal{X}^j_{\text{terminal}}. && && \text{(2.5f)}
\end{aligned}
$$

The admissible set when a fault has occurred is therefore given by

$$\mathbb{Z}^j_N = \{x_0, \mathbf{u} \mid \exists \mathbf{x} \text{ such that } (\mathbf{x}, \mathbf{u}) \text{ satisfies } (2.5\text{b}) - (2.5\text{f})\}. \tag{2.6}$$

The feasible set $\mathbb{X}^j_N$ denoting the set of states in which the MPC problem is feasible for a fault in actuator j is therefore defined as

$$\mathbb{X}^j_N = \left\{x \mid \text{such that } (\mathbf{x}, \mathbf{u}) \in \mathbb{Z}^j_N\right\}, \tag{2.7}$$

where it is assumed that the terminal set in (2.5f) is control invariant, so that the problem is feasible for all time. As for the nominal case, this set can be interpreted as the maximal control invariant set by means of the terminal set $\mathcal{X}^j_{\text{terminal}}$ and horizon N. From this, it is clear that the safety set needs to be defined as

$$\mathbb{S}_j = \mathbb{X}^j_N. \tag{2.8}$$

Then, by making sure that the system operates in $\mathbb{S}_j$ when the fault in actuator j occurs, the MPC problem with the fault present will be feasible for all time. The next section describes a general framework for the computation of the control invariant sets in order to represent the safety sets.

2.4 Computing control invariant sets

The previous section defined the safety set to be the feasible set for the MPC problem when a fault has occurred, which can be thought of as the maximal control invariant set with respect to the constraints introduced by the fault. This section describes a general framework for the computation of the aforementioned set. An approach for directly computing the set of states for which the MPC problem is feasible, would be to directly project (2.2) onto $\mathbb{R}^n$. That is, to eliminate u from the constraints, and represent the same constraints by x. One would then find the set of states for which the constraints are satisfied for the prediction horizon. With this approach, the computation of the feasible set relies essentially on the efficiency of projection algorithms. However, these projections tend to become computationally demanding as the prediction horizon increases (Scibilia et al., 2011). As such, most methods rely on iterative approaches where less demanding operations are repeated until the set is found, see e.g. Scibilia et al. (2011); Kerrigan and Maciejowski (2000a); Athanasopoulos and Bitsoris (2010)

The focus of this chapter is to describe a general iterative framework to develop such algorithms, rather than a detailed description of the algorithms themselves, which would require the introduction of several additional theoretic concepts, and is therefore outside the scope of this thesis. However, most algorithms in the literature rely on this framework and it is therefore of great value for the development of algorithms. The following definition is essential for the development of the aforementioned framework.

Definition 2.7 (One-step set (Blanchini, 1994))**.** The non-empty set $\mathcal{Q}(\Omega)$ is defined as the set of states in $\mathbb{R}^n$ for which an admissible control input exists which will drive the system to Ω in one step, i.e.

$$\mathcal{Q}(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} \text{ such that } f(x_k, u_k) \in \Omega\}. \tag{2.9}$$

The key ingredients for implementing the iterative algorithms are procedures for computing:

- The one-step set $\mathcal{Q}(\cdot)$ (Definition 2.7);
- The intersection of two sets;
- Equality of sets or whether a set is a subset of another.

Several methods exist for these computations, and they often vary for different algorithms. However, they are relatively straightforward for LTI systems subject to constraints on the states and control inputs (Kerrigan and Maciejowski, 2000a).

The one-step set may be computed using projection methods. Projection algorithms were mentioned earlier to be computationally demanding. However, note that this is a projection for a single timestep rather than for the whole prediction horizon, and the procedure is therefore less demanding. A common way of computing this projection is by Fourier-Motzkin elimination, which is the equivalent of Gaussian elimination for solving a set of linear inequalities, see e.g. (Keerthi and Gilbert, 1987). Another commonly used approach for computing the one-step set is via Minkowski summation, see e.g. Scibilia et al. (2011). The intersection of two sets as well as equality and subsets tests can be done by the methods layed out in Fukuda (2004).

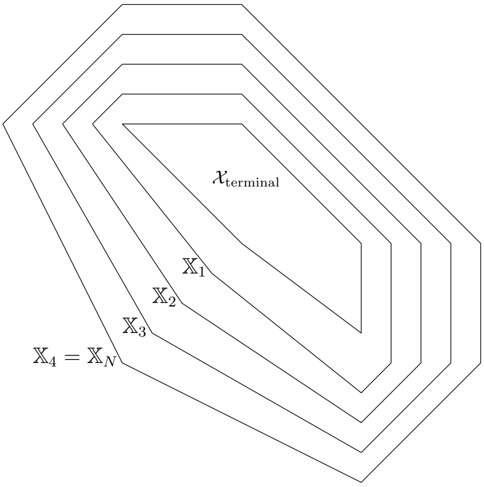

$\mathcal{X}_{\text{terminal}}$
$\mathbb{X}_1$
$\mathbb{X}_2$
$\mathbb{X}_3$
$\mathbb{X}_4 = \mathbb{X}_N$

Figure 2.1: Illustration of the convergence of Algorithm 2.1 to the feasible set for a prediction horizon of $N = 4$.

Given that methods for the aforementioned computations are available, the feasible set, i.e. the maximal control invariant set for the system by means of the constraints, can be computed by the following algorithm

Algorithm 2.1 (Scibilia et al. (2011))**.** The feasible set, $\mathbb{X}_N$, of a system can be computed via the following iterative procedure:

1. Initialize the set $\mathbb{X}_0 = \mathcal{X}_{\text{terminal}}$ and set $i = 0$.

2. Compute the one step set $\mathcal{Q}(\mathbb{X}_i)$.

3. Compute the intersection $\mathcal{Q}(\mathbb{X}_i) \cap \mathbb{X}$ and set $\mathbb{X}_{i+1} = \mathcal{Q}(\mathbb{X}_i) \cap \mathbb{X}$.

4. If i=N or $\mathbb{X}_i = \mathbb{X}_{i+1}$ then terminate and set $\mathbb{X}_N = \mathbb{X}_i$. Else, set i=i+1, go to step 2.

Figure 2.1 illustrates the idea in Algorithm 2.1. The feasible set is of great importance when it comes to defining operating regions to guarantee MPC feasibility. However, the representation of the set often becomes complex for systems with many states. Several approximation methods which aim to approximate the set using less complex presentations have emerged in the literature. The objective is to find simpler polytopic representations in order to reduce computational load. The reader is referred to Scibilia et al. (2011) for a more detailed explanation.

2.5 Numerical illustrative examples

This section contains computations of the safety sets for the examples introduced in Chapter 1, based on the theory described in this chapter. The sets are computed as maximal control invariant sets using the Multi Parametric Toolbox 3 (MPT3)[1].

2.5.1 Example with two states, two inputs and one dropout

Using MPT3, the resulting safety set is defined by

$$\mathbb{S}_2 = \left\{ x \ : \begin{bmatrix} 0.6202 & 0.7845 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 1.7060 \\ 0 \\ 0 \end{bmatrix} \right\}, \tag{2.10}$$

and is plotted in Figure 2.2 together with the nominal state constraints and the feasible set for the healthy system. Due to the fact that it is the most powerful actuator that has a dropout, the feasible set shrinks considerably. It is critical that the system operates in the set $\mathbb{S}_2$ when the dropout of the second actuator occurs. The failure to steer the system into this set before the fault occurs will result in destabilization of the process.

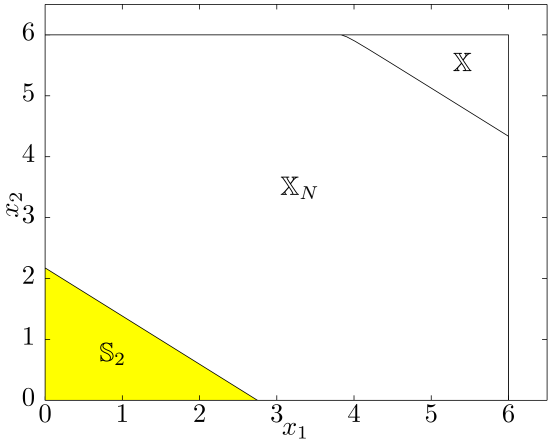


Figure 2.2: Numerical example of the nominal feasible set and the safety set for a single actuator dropout.

2.5.2 Example with two states, three inputs and two dropouts

This example considers a situation with multiple actuator dropouts. It is therefore necessary to compute multiple safety sets. A safety set for the first fault, i.e. the

[1] Available for download at `http://people.ee.ethz.ch/~mpt/3/`

dropout of actuator two, as well as the safety set for a situation where both actuator one and two are rendered useless, is considered.

The safety set for the dropout of actuator two, $\mathbb{S}_2$, is computed to be

$$\mathbb{S}_2 = \left\{ x \ : \begin{bmatrix} 0.7465 & 0.6654 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 7.6579 \\ 0 \\ 0 \end{bmatrix} \right\}. \tag{2.11}$$

When in addition actuator one has a dropout, the safety set becomes

$$\mathbb{S}_{12} = \left\{ x \ : \begin{bmatrix} 0.7465 & 0.6654 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 2.4721 \\ 0 \\ 0 \end{bmatrix} \right\}. \tag{2.12}$$

As can be seen in Figure 2.3, the feasible set shrinks considerably in the case of multiple dropouts. It is necessary that the system operates in the set $\mathbb{S}_2$ when the first fault occurs, and in the set $\mathbb{S}_{12}$ when the additional fault occurs, in order to avoid an unstable system response.

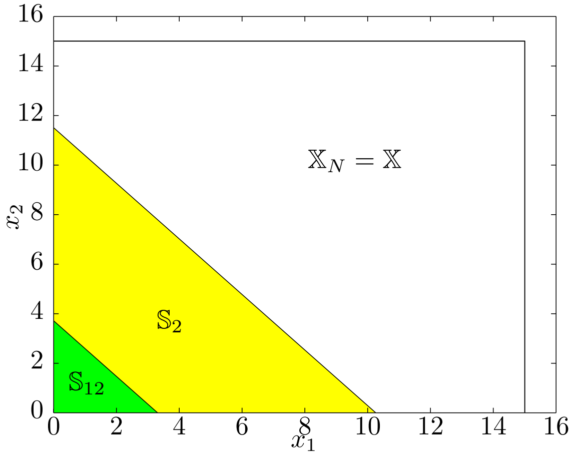


Figure 2.3: Numerical example of multiple safety sets for multiple actuator dropouts.

Chapter 3

Soft Constraints and Penalty Functions

In Chapter 1.4.2, penalty functions were mentioned to be an important ingredient for the proactive fault-tolerant scheme described in this thesis. This chapter describes the theory of penalty functions, and in particular *exact* penalty functions. The first section introduces the main concept and contains a brief background review, while the following sections focus on the use of penalty functions in MPC. A method for computing a lower bound for the penalty weight to guarantee exactness is presented, and the last section contains computations for the recurring illustrative examples.

3.1 Introduction

Penalty functions have been a part of the literature on constrained optimization for decades. Their most common use is in methods for solving nonlinear and nonsmooth optimization problems (Fletcher, 1987), as well as for handling infeasibility issues in MPC (Kerrigan and Maciejowski, 2000b), which is the main focus of this chapter. In MPC, a problem occurs if a disturbance or a change in operating conditions drives the plant into a state where the control problem is infeasible and the MPC is not able to compute a new control input. A systematic method for dealing with infeasibility is by the use of soft constraints, where the constraints are allowed to be violated, but the violation is penalized in the cost function.

It is important to distinguish input constraints from state constraints. The input constraints arise from physical limitations of the actuators such as valve saturations, and it is always advantageous to enforce them in the control law. State constraints, on the other hand, are often not due to physical limitations, but rather they are included to maintain the process in a desired operating regime (Scokaert and Rawlings, 1999). As will be seen, this is clearly the case for the defined safety set for the proactive FTEMPC scheme introduced in Section 2.3, as this is a desired (and also necessary) operating region for when a known fault occurs.

Additionally, it is desirable that the solution to the soft-constrained problem is the same as the solution to the original hard-constrained problem, if the latter is feasible. This is important in order to avoid unnecessary violations. The theory of exact penalty functions can be used to compute a lower bound for the constraint violation weight such that equality is guaranteed (Fletcher, 1987). A method for this computation is described in Section 3.4.

3.2 Exact penalty functions

This section briefly describes the basic theory of penalty functions, and states the basic well-known result that guarantees exactness of a penalty function, thereby ensuring that the solutions to the soft- and hard constrained problem are equal.

Consider the constrained optimization problem:

$$\begin{aligned} \min_{x} \quad & V(x) \\ \text{s.t.} \quad & G(x) \leq 0, \end{aligned} \tag{3.1}$$

where $V(x)$ is the cost function, and $G(x)$ defines the constraints. The corresponding non-smooth penalty function minimization is:

$$\min_{x} V(x) + \mu \left\| G(x)^{+} \right\|, \tag{3.2}$$

where μ is the constraint violation penalty weight and $G(x)^{+} = \max(G(x)_i, 0)$. The vector $G(x)^{+}$ contains the values of the constraint violations for a given x.

In the penalty function problem (3.2), the constrained optimization problem has been reformulated to an unconstrained problem where violations of the original constraints are allowed, but penalized. These penalty functions are exact in the sense that the solution to the original minimization problem is the same as for the reformulated penalty function minimization for sufficiently large μ.

The objective is thus to find a lower bound for the penalty parameter that ensures exactness. First, the concept of the dual norm needs to be introduced, which is an essential component in the theory of exact penalty functions. For any given norm $\|\cdot\|$, there is a corresponding norm $\|\cdot\|_{\mathrm{D}}$ that is called the dual norm and is defined by (Kerrigan and Maciejowski, 2000b)

$$||\nu||_{\mathrm{D}} = \max_{||\lambda|| \leq 1} \nu' \lambda. \tag{3.3}$$

It can be shown that the the dual of $\|\cdot\|_1$ is $\|\cdot\|_\infty$ and vice versa. $\|\cdot\|_2$ is its own dual. Using the concept of the dual norm, the following theorem from Fletcher (1987) explains the criteria for a penalty function to be exact.

Theorem 3.1 (Exact penalty function)**.** *If the penalty weight $\mu \geq \|\lambda^*\|_D$ and $G(x) \leq 0$, then the solution x^* to* (3.2) *is equal to the solution of* (3.1)*.*

See (Fletcher, 1987, Chapter 14) for proof.

Nocedal and Wright (2006) describes this in the following way: At a solution of the optimization problem, x^*, any move into the infeasible region is penalized enough so that it produces an increase in the cost function to a value greater than the value at x^*, thereby forcing the minimum to lie at x^*. This means that the penalty term in the objective function will increase the overall cost more than what can be gained in the original cost term by violating the constraint.

This result also holds true if different penalty parameters are used for each constraint, thus the penalty parameter needs to be larger than the Lagrangian multiplier for its respective violated constraint, see Janesch and Santos (1997). This will be useful when computing a lower bound for the penalty weight in Section 3.4.

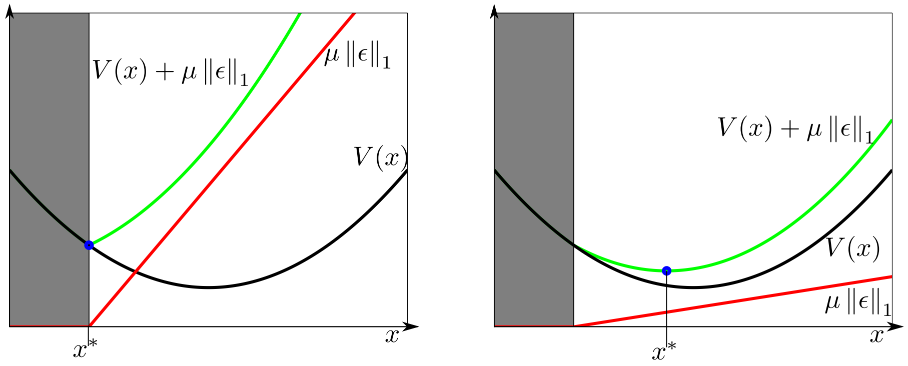
3.3 Soft constrains and penalty functions in MPC

This section presents the use of soft constraints and penalty functions in MPC. A potential problem with the penalty minimization problem (3.2) from the previous section is that it is non-smooth, which is not always easy to solve. One can overcome this by introducing slack variables, which is a way of rewriting the penalty function formulation to a smooth formulation that still yields the same results. Problem (3.2) is rewritten to:

$$\begin{aligned} \min_{x,\epsilon} \quad & V(x) + \mu \|\epsilon\| \\ \text{s.t.} \quad & G(x) \leq \epsilon \\ & \epsilon \geq 0, \end{aligned} \tag{3.4}$$

where ϵ are the slack variables representing the constraint violations, i.e. $\epsilon = 0$ if the constraints are satisfied. As can be seen in the optimization problem (3.4), non-zero values of ϵ are penalized in the objective function. Figure 3.1 illustrates the concept of an exact penalty function for this reformulation with a quadratic cost function and an ℓ_1-penalty function. As can be seen from the figure, a large enough penalty value produces an increase in the cost function when constraints are violated. However, for a smaller penalty parameter, the violation is not penalized enough to produce this increase, and the problem has an optimal value outside the feasible area.

For compactness of notation, a recasted compact form of the MPC problem is used,



(a) A large enough value of μ such that the penalty function is exact.

(b) A too small value of μ such that the penalty function is inexact.

Figure 3.1: Illustration showing the comparison between an exact and an inexact penalty function for a quadratic cost. The gray area represents the feasible region.

as described in Appendix A. The soft constrained MPC problem is then stated as:

$$\begin{aligned} \min_{\mathbf{u},\epsilon} \quad & l_c(x_0, \mathbf{u}) + \phi(\epsilon) \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0 + \theta(\epsilon) \\ & \epsilon \geq 0, \end{aligned} \tag{3.5}$$

where $\phi(\epsilon) : \mathbb{R}^{d_\epsilon} \mapsto \mathbb{R}$ is the penalty function, and $\theta(\epsilon) : \mathbb{R}^{d_\epsilon} \mapsto \mathbb{R}^q$ is the constraint function. The scalar d_ϵ denotes the number of rows in ϵ, i.e. $\epsilon \in \mathbb{R}^{d_\epsilon}$ and the scalar q denotes the number of rows in G, i.e. the number of constraints in the problem.

The penalty functions that appear frequently in the MPC literature are the ℓ_1-penalty function, the ℓ_∞-penalty function and the quadratic penalty function:

$$\ell_1(x, \epsilon) = \mu \left\|\epsilon\right\|_1, \tag{3.6}$$
$$\ell_\infty(x, \epsilon) = \mu \left\|\epsilon\right\|_\infty, \tag{3.7}$$
$$\ell_2^2(x, \epsilon) = \mu \left\|\epsilon\right\|_2^2. \tag{3.8}$$

These are briefly presented in the following sections in order to give an overview and to introduce their drawbacks and advantages. The following notations are used: The matrix I_q denotes the identity matrix with q diagonal elements, and the vector $\mathbf{1}_q$ denotes a vector of ones in $\mathbb{R}^q$.

ℓ_1-penalty function

The ℓ_1-penalty function penalizes the sum of the constraint violations over the prediction horizon, and is included in the MPC formulation in the following way:

$$\begin{aligned}\min_{\mathbf{u},\epsilon} \quad & l_{\mathrm{c}}(x_0, \mathbf{u}) + \mu \left\|\epsilon\right\|_1 \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0 + I_q\epsilon \\ & \epsilon \geq 0,\end{aligned} \tag{3.9}$$

where ϵ is a vector of q rows, i.e. $d_\epsilon = q$. The multiplication of ϵ with the identity matrix will become clear in Section 3.4. Note, however, that this does not change the problem. The ℓ_1 norm penalty function increases the number of decision variables in the optimization problem by the number of constraints that are relaxed, and will therefore increase computational load. Although it is shown in Rao et al. (1998) that the addition of the ℓ_1 optimization variables can be handled at virtually no additional computational cost if the problem structure is utilized in the solver.

ℓ_∞-penalty function

The ℓ_∞ penality function penalizes the value of the the largest constraint violation that is predicted to occur over the prediction horizon and is included in the MPC optimization problem in the following way:

$$\begin{aligned}\min_{x_0,\mathbf{u}} \quad & l_{\mathrm{c}}(\mathbf{u}, \epsilon) + \mu \left\|\epsilon\right\|_\infty \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0 + \epsilon \\ & \epsilon \geq 0,\end{aligned} \tag{3.10}$$

where ϵ is a vector with q rows. Note that this is a non-smooth optimization problem due to the $\left\|\epsilon\right\|_\infty$ term. This can be rewritten to a smooth form by (Maciejowski, 2002):

$$\begin{aligned}\min_{x_0,\mathbf{u}} \quad & l_{\mathrm{c}}(x_0, \mathbf{u}) + \mu\epsilon \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0 + \mathbf{1}_q\epsilon \\ & \epsilon \geq 0,\end{aligned} \tag{3.11}$$

where ϵ is a scalar, i.e. $d_\epsilon = 1$. The ℓ_∞-penalty norm only increases the number of decision variables in the optimization problem by 1. For this reason, ℓ_∞ norm penalty functions are often preferred. However, the ℓ_∞ norm can result in unexpected behavior and poor performance if it is used to soften an output constraint for which there is an inverse response (Hovd and Stoican, 2014). A method for minimizing this problem is provided in Hovd and Braatz (2001) using time dependent weights. However, this will not be investigated further in this thesis.

ℓ_2^2 penalty function

The quadratic penalty function penalizes the squared value of the constraint violation:

$$\begin{aligned} \min_{\mathbf{u},\epsilon} \quad & l_{\mathrm{c}}(x_0,\mathbf{u}) + \mu \left\|\epsilon\right\|_2^2 \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0 + I_q\epsilon \\ & \epsilon \geq 0, \end{aligned} \tag{3.12}$$

where ϵ is a vector of q rows. A drawback of the quadratic penalty function is that if the constraints are active, then for all finite values of μ this formulation will result in them being violated to some extent, even if such violation is not necessary. Thus, it is not possible to make the quadratic penalty function exact. This is due to the fact that the penalty function formulation (3.2) will be smooth with a quadratic penalty function and it is the non-smoothness of the penalty function which allows it to be exact (Kerrigan and Maciejowski, 2000b). Thus, the essential property of the ℓ_1 and ℓ_∞-penalty functions, which allows them to be made exact, is that they have a discontinuity in slope at the zero value of the slack variables. To this end, Theorem 3.1 yields an important result for a penalty function to be exact when using either the ℓ_1 or the ℓ_∞-penalty function:

- By using the penalty function $\phi(\epsilon) = \mu\left\|\epsilon\right\|_1$, the penalty function is exact when $\mu \geq \max\limits_{\lambda} \left\|\lambda\right\|_\infty$.
- By using the penalty function $\phi(\epsilon) = \mu\left\|\epsilon\right\|_\infty$, the penalty function is exact when $\mu \geq \max\limits_{\lambda} \left\|\lambda\right\|_1$.

Another important property of the ℓ_1-penalty function is that, because it penalizes the sum of constraint violations over the whole horizon, an exact penalty parameter will in fact produce a minimal time response of the controller for steering the system into the feasible region. This is elaborated on in Chapter 4.

As described, a sufficiently high value of the linear term in the penalty function will ensure that the penalty function is exact. However, a too large term is generally not desirable, since it may lead to unnecessarily violent control which may be harmful to the actuators. Therefore one would wish to find the minimal values that guarantee exactness, which corresponds to the largest dual norm of the Lagrangian multiplier as defined in Theorem 3.1. This is a non-convex optimization problem which in general has been considered intractable, and is not straightforward to compute (Hovd and Stoican, 2014). The next section presents a method based on two recently published papers.

3.4 Computing a lower bound for the penalty weight

It is desirable that the solution to the soft-constrained MPC problem and the solution to the original hard-constrained MPC problem only differs if the original problem is infeasible. Section 3.2 provided a criteria for the penalty parameter value for this to hold, i.e. that the value of the penalty parameter needs to be greater than the largest value of the dual norm of the Lagrangian multiplier. However, in MPC, the Lagrangian multipliers are dependent on the current state of the system (Kerrigan and Maciejowski, 2000b). It is therefore necessary to compute the Lagrangian multipliers for the whole feasible set of the hard-constrained problem. As noted in Kerrigan and Maciejowski (2000b), a naive and impractical solution would be to grid the state space region of interest and compute the optimal Lagrange multipliers at each point. This would be a computationally heavy approach, and does not guarantee that a solution is found. Kerrigan and Maciejowski (2000b) propose a method that solves a series of linear programs to find a lower bound, however, this approach makes some heavy assumptions for the system constraints. This section presents a method based on two recently published papers, Hovd (2011); Hovd and Stoican (2014), that relaxes these assumptions.

3.4.1 Preliminaries

This section gives the necessary preliminaries and notations used for solving the optimization problem that yields the largest Lagrangian multipliers used for designing penalty parameters. The method formulates the search for the largest Lagrangian multiplier as a *bi-level program* reformulated to a *mixed integer linear program* (MILP). Additionally, the approach relies on the theory of *polyhedral norms* (Anderson and Osborne, 1976) in order to provide a general framework for the computation and thereby avoiding separate optimization problems for each type of penalty function. The basics of bi-level programming and polyhedral norms are presented in the following sections.

Bi-level programming

Bi-level optimization is a type of optimization where the constraints of the main problem involve the solution solution of another optimization problem. The notations in this section are taken from Hovd and Stoican (2014). A general form of this type of problem is the following:

$$\min_{y} \quad V_U(y,z) \tag{3.13a}$$

$$\text{s.t.} \tag{3.13b}$$

$$G_{UI}(y,z) \leq 0 \tag{3.13c}$$

$$G_{UE}(y,z) = 0 \tag{3.13d}$$

$$\min_{z} \; V_L(y,z) \tag{3.13e}$$

$$\text{s.t.} \tag{3.13f}$$

$$G_{LI}(y,z) \leq 0 \tag{3.13g}$$

$$G_{LE}(y,z) = 0, \tag{3.13h}$$

where $y \in \mathbb{R}^{n_1}$ and $z \in \mathbb{R}^{n_2}$. The variables of problem (3.13) are divided into two classes, namely the *upper-level variables* y, and the *lower-level variables* z. The functions $V_U : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}$ and $V_L : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}$ are the *upper-level* and *lower-level objective functions* respectively. The functions $G_{UI} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}^{m_{11}}$, $G_{UE} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}^{m_{12}}$ are the *upper-level constraints* and $G_{LI} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}^{m_{21}}$, $G_{LE} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \mapsto \mathbb{R}^{m_{22}}$ are the *lower-level constraints* (Colson et al., 2005).

When the lower-level problem is *convex* and regular, it can be replaced by its Karush-Kuhn-Tucker (KKT) conditions (see Appendix B). Replacing the lower level problem of (3.13) by its KKT-conditions yields:

$$\begin{aligned}
\min_{y,z,\lambda,\gamma} \quad & V_U(y,z) && \text{(3.14a)}\\
& \text{s.t.} && \text{(3.14b)}\\
& G_{UI}(y,z) \leq 0 && \text{(3.14c)}\\
& G_{UE}(y,z) = 0 && \text{(3.14d)}\\
& \lambda \geq 0 && \text{(3.14e)}\\
& G_{LI}(y,z) \leq 0 && \text{(3.14f)}\\
& G_{LE}(y,z) = 0 && \text{(3.14g)}\\
& \lambda \times G_{LI}(y,z) = 0 && \text{(3.14h)}\\
& \nabla_z \mathscr{L}(y,z,\lambda,\gamma) = 0, && \text{(3.14i)}
\end{aligned}$$

where $\mathscr{L}(y,z,\lambda,\gamma)$ is the Lagrangian of the lower level problem and λ is the vector of Lagrangian multipliers for the lower level problem. The × symbol indicates that the kth element of the vector λ of Lagrangian multipliers is multiplied with the kth constraint in the lower level inequality constraints. Notice that this problem is non-convex due to the complementary condition (3.14h) where two decision-variables are multiplied. Fortuny-Amat and McCarl (1981) suggests a technique to reformulate the problem in to a convex mixed integer linear program using binary variables. Applying this approach gives:

$$\begin{aligned}
\min_{y,z,\lambda,\gamma,s} \quad & V_U(y,z) && \text{(3.15a)}\\
\text{s.t.} \quad & && \text{(3.15b)}\\
& G_{UI}(y,z) \leq 0 && \text{(3.15c)}\\
& G_{UE}(y,z) = 0 && \text{(3.15d)}\\
& \lambda \geq 0 && \text{(3.15e)}\\
& \lambda \leq Ms && \text{(3.15f)}\\
& G_{LI}(y,z) \leq 0 && \text{(3.15g)}\\
& G_{LE}(y,z) = 0 && \text{(3.15h)}\\
& G_{LI}(y,z) \geq -M(1-s) && \text{(3.15i)}\\
& \nabla_z \mathscr{L}(y,z,\lambda,\gamma) = 0 && \text{(3.15j)}\\
& s \in \{0,1\}^l, && \text{(3.15k)}
\end{aligned}$$

where M is a sufficiently large scalar and l is the number of inequality constraints in the lower level problem. Note that the complimentary condition (3.14h) is replaced by conditions (3.15f) and (3.15i). Problems (3.14) and (3.15) yield the same result through suitable combinations of binary variables $s \in \{0,1\}^l$, which is an optimization variable. $s_i = 1$ represents the constraint $G_{LI}(y,z)_i$ as active and λ_i inactive. $s_i = 0$ yields $G_{LI}(y,z)_i$ inactive and λ_i active. Thus, the non-convex complementary condition has been rewritten as two convex constraints. This result will be used to formulate the search for the maximum Lagrangian multiplier as a MILP.

Polyhedral norms

It is desirable to state the optimization problem used to compute the largest Lagrangian multipliers using a general formulation, rather than separate formulations for each norm (e.g. seperate for ℓ_1 and ℓ_∞). For this, Hovd and Stoican (2014) proposes to use polyhedral norms. The soft constraint and the penalty function is restated in the following more general way:

$$\phi(\epsilon) = F_\epsilon^T \epsilon, \tag{3.16}$$

$$\theta(\epsilon) = G_\epsilon \epsilon, \tag{3.17}$$

where F_ϵ^T and G_ϵ are designed based on the penalty function that is considered, most typically ℓ_1 or ℓ_∞. As will be shown, by using the more general and flexible form, there is no need for separate formulations for the different norms, and they can all be included in the same framework. The definition of the polyhedral norm becomes useful

Definition 3.1 (Blanchini (1995))**.** Having a polyhedral set given in the form $\{x : (Gx)_i \leq 1,\ i = 1 \ldots n\}$, its associated polyhedral norm is defined as $\Psi(G,x) = \max\limits_{i=1\ldots n} (Gx)_i$ where n denotes the number of rows in matrix G.

Additionally, as noted in Hovd and Stoican (2014), one can without loss of generality state F_ϵ from Equation (3.16) as

$$F_\epsilon = \mu \mathbf{1}_{d_\epsilon}. \tag{3.18}$$

The criteria for an exact penalty function from Theorem 3.1 can now be stated as follows:

Corollary 3.1. *(Hovd and Stoican, 2014) Functions $\phi(\epsilon)$ and $\theta(\epsilon)$ written as in* (3.16) *and* (3.17) *assure an exact correspondence between the hard constrained and the soft constrained problem if*

$$\mu \geq \max_{\lambda} \Psi(G_\epsilon^T, \lambda) \tag{3.19}$$

See Hovd and Stoican (2014) for proof. This condition includes the classical ℓ_1 and ℓ_∞ conditions for exact hard constraints described in Section 3.2, i.e., that the weight μ on the linear term of the penalty function has to be larger than the maximal value of the dual norm of the Lagrangian multipliers of the corresponding hard-constrained optimization problem:

- For ℓ_1-penalty: $d_\epsilon = q$ and by taking $\theta(\epsilon) = I_q \epsilon$ one has that $\phi(\epsilon) = \mu \mathbf{1}_{d_\epsilon}^T \epsilon = \mu \|\epsilon\|_1$ where $\mu \geq \max\limits_{\lambda} \Psi(I_q^T, \lambda) = \max\limits_{\lambda} \|\lambda\|_\infty$;

- For ℓ_∞ penalty: $d_\epsilon = 1$ and by taking $\theta(\epsilon) = \mathbf{1}_q \epsilon$ one has that $\phi(\epsilon) = \mu\epsilon = \mu \|\epsilon\|_\infty$ where $\mu \geq \max\limits_{\lambda} \Psi(\mathbf{1}_q^T, \lambda) = \max\limits_{\lambda} \|\lambda\|_1$.

Thus, by defining the constraint function $\theta(\epsilon)$ and the criteria in Corollary 3.1 with polyhedral norms, the criteria in Theorem 3.1 is implicitly taken into account. Also note that, as stated in Section 3.2, one only needs to find the Lagrangian multipliers for the constraints that are softened. This may be done by setting the elements i G_ϵ corresponding to a non-softened constraint to zero, and the elements corresponding to a softened constraint to one. Notice then that slack is only allowed on the allowed softened constraints in (3.5), and also that $\max\limits_{\lambda} \Psi(G_\epsilon^T, \lambda)$ from Corollary 3.1 will ignore the non-softened constraints. Hence, only the Lagrangian multipliers corresponding to the softened constraints are evaluated in the criteria in Corollary 3.1.

3.4.2 Using bi-level and mixed integer programming

Hovd (2011) introduces an approach for computing the maximum Lagrangian multipliers of the MPC problem in order to find a lower bound for the penalty parameter that satisfies Theorem 3.1. However, this approach is computationally demanding and Hovd and Stoican (2014) further develops this approach to increase numerical "quality" and computation speed. This section begins by describing the first approach in order to illustrate the basic idea, and continues to describe the improved and more efficient approach.

The objective is to find a lower bound on the penalty parameter μ that guarantees exactness of the penalty function. This can be done by solving the bi-level program:

$$\max_{x_0} \quad \Psi(G_\epsilon^T, \lambda) \tag{3.20a}$$

$$\text{s.t.} \tag{3.20b}$$

$$\min_{\mathbf{u}} l_c(x_0, \mathbf{u}) \tag{3.20c}$$

$$\text{s.t.} \tag{3.20d}$$

$$G\mathbf{u} \leq W + Ex_0, \tag{3.20e}$$

where λ is the vector of Lagrangian multipliers for the MPC problem and $\mathbf{u}$ is the solution of the MPC problem. Note that this problem is not well-defined until the KKT-conditions for the MPC problem replace the lower level problem. This yields:

$$\max_{x_0,\mathbf{u},\lambda} \quad \Psi(G_\epsilon^T,\lambda) \tag{3.21a}$$

$$\text{s.t.} \tag{3.21b}$$

$$G\mathbf{u} - W - Ex_0 \leq 0 \tag{3.21c}$$

$$\lambda \geq 0 \tag{3.21d}$$

$$\nabla_{\mathbf{u}} l_c(\mathbf{u},x_0) + G^T\lambda = 0. \tag{3.21e}$$

$$\lambda \times (G\mathbf{u} - W - Ex_0) = 0. \tag{3.21f}$$

As mentioned in the previous chapter, this problem is non-convex. By applying the same reformulation as described earlier, the problem becomes:

$$\max_{x_0,\mathbf{u},\lambda,s} \quad \Psi(G_\epsilon^T,\lambda) \tag{3.22a}$$

$$\text{s.t.} \tag{3.22b}$$

$$G\mathbf{u} - W - Ex_0 \leq 0 \tag{3.22c}$$

$$G\mathbf{u} - W - Ex_0 \geq -M(1-s) \tag{3.22d}$$

$$\lambda \geq 0 \tag{3.22e}$$

$$\lambda \leq Ms \tag{3.22f}$$

$$\nabla_{\mathbf{u}} \ell(\mathbf{u},x_0) + G^T\lambda = 0 \tag{3.22g}$$

$$s \in \{0,1\}^q. \tag{3.22h}$$

Although the KKT conditions for the MPC problem (the lower-level problem) uniquely determine the optimal **u**, they do not uniquely determine the Lagrangian multipliers λ. As noted in Hovd and Stoican (2014), this will result in unnecessarily large λs, bounded only by M. As earlier described, the goal is to find the smallest λs that fulfill the KKT conditions for the MPC problem. In Hovd (2011), this is done by adding an extra minimization problem to (3.22).

3.4.3 Adding an extra minimization problem

An alternative approach to handle the non-uniqueness of the Lagrangian multipliers is to add an extra minimization problem, which will compute unique Lagrangian multipliers as described in Hovd (2011). Consider again the problem (3.22). Following the aforementioned approach yields:

$$
\begin{aligned}
\max_{x_0,\mathbf{u},\lambda,s} \quad & \Psi(G_\epsilon^T,\lambda) && \text{(3.23a)}\\
\text{s.t.} \quad & && \text{(3.23b)}\\
& G\mathbf{u} - W - Ex_0 \leq 0 && \text{(3.23c)}\\
& G\mathbf{u} - W - Ex_0 \geq -M(1-s) && \text{(3.23d)}\\
& \min_{\lambda} \ \Psi(G_\epsilon^T,\lambda) && \text{(3.23e)}\\
& \text{s.t.} && \text{(3.23f)}\\
& \lambda \geq 0 && \text{(3.23g)}\\
& \lambda \leq Ms && \text{(3.23h)}\\
& \nabla_{\mathbf{u}}\ell(x_0,\mathbf{u}) + G^T\lambda = 0 && \text{(3.23i)}\\
& s \in \{0,1\}^q, && \text{(3.23j)}
\end{aligned}
$$

which can be solved by following the previously described procedure for recasting a bi-level program to a MILP. However, this approach introduces many additional variables during the recasting procedure, which will increase computational time and might also decrease numerical quality. The next section describes an approach originally published in Hovd and Stoican (2014), which circumvents this issue.

3.4.4 Adding an explicit LICQ constraint

As described, the Lagrangian multipliers are not necessarily uniquely defined for the MILP (3.22). In the previous section, this was solved by adding additional minimization subproblems such that the minimum multiplier is selected from the space of available solutions. However, the reformulation introduces many additional decision variables, and the problem can therefore become computationally heavy for large systems with long prediction horizons. Hovd and Stoican (2014) builds on this approach and introduces concepts that will improve computation speed and increase numerical accuracy by explicitly defining the LICQ condition, defined below, as a constraint in the optimization problem (3.22). This will guarantee uniqueness of the solution.

Definition 3.2. (Nocedal and Wright, 2006, Chapter 12). For an active set, the LICQ holds if the set of active constraint gradients is linearly independent.

As long as the set of active constraint gradients respects the LICQ, uniqueness of the solution is guaranteed. The approach in this section exploits the structure of the constraint matrix G, so that only a subset of constraints, which respect the LICQ, are chosen in the optimization problem.

Consider the constraint matrix G, and note that at timestep k, only x_0 and the sequence of inputs $u_0,\ldots,u_k$ appear in the constraints. This means that G has a lower triangular structure, see Figure 3.2. Each block R_k describes the kth order

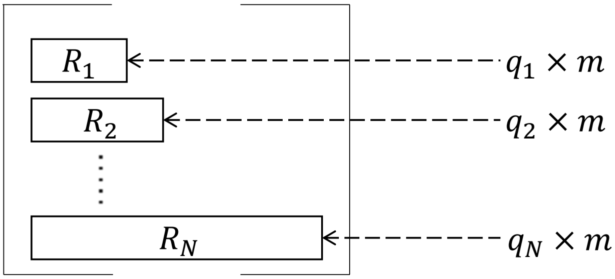


Figure 3.2: The constraint matrix, G. From Hovd(2014)

constraints and has q_k rows and km columns, where m denotes the number of inputs in the system. This yields the following result that guarantees the LICQ to hold for different subsets of active constraints (Hovd and Stoican, 2014). Let i_k denote the number of constraints selected from each block R_k. Then the conditions

$$0 \le i_k \le \min(q_k, km), \ \sum_{j=0}^{k} i_j \le km, \quad \forall k = 1, \dots, N. \tag{3.24}$$

define all the selections of constraints which can satisfy the LICQ. The reasoning for the first condition is as follows: from a collection of 'y' rows where only the first 'x' elements in each row are non-zero, at most '$\min(x, y)$' rows can be selected and still be linearly independent. For the second condition note that from the first q_1 rows one can select at most m, from the first $q_1 + q_2$ rows one can select at most $2m$ and from the first $q_1 + \cdots + q_N$ rows one can select at most Nm rows. See Hovd and Stoican (2014) for a more detailed proof.

Finally, including the LICQ condition (3.24) in the optimization problem (3.22) yields:

$$\max_{j=1\dots d_\epsilon} \ \max_{x_0, \mathbf{u}, \lambda, s} \ \Psi(G_\epsilon^T, \lambda) \tag{3.25a}$$
$$\text{s.t.} \tag{3.25b}$$
$$G\mathbf{u} - W - Ex_0 \le 0 \tag{3.25c}$$
$$G\mathbf{u} - W - Ex_0 \ge -M(1-s) \tag{3.25d}$$
$$\lambda \ge 0 \tag{3.25e}$$
$$\lambda \le Ms \tag{3.25f}$$
$$\nabla_{\mathbf{u}} l(\mathbf{u}, x_0) + G^T \lambda = 0 \tag{3.25g}$$
$$0 \le \sum_{i=\tau_{k-1}}^{t_k} s_i \le \min(q_k, km), \ \sum_{i=1}^{\tau_k} s_i \le km \tag{3.25h}$$
$$s \in \{0,1\}^q, \tag{3.25i}$$

where $\tau_k = q_1 + \cdots + q_k$ for any $k = 1, \dots, N$ and one has to solve d_ϵ subproblems to solve the overall problem. The first max operator comes from the definition of the polyhedral norm and the order of the max operators are switched, see Hovd and Stoican (2014). By solving problem (3.25) one can guarantee that the solution satisfies LICQ due to the inclusion of the LICQ constraint (3.25h) and thus the solution is unique.

The procedure for computing the maximum Lagrangian multipliers may be summarized as follows:

1. Recast the MPC problem in to compact form, i.e. design matrices H, F, G, E and W and make sure that G is in lower block triangular form;

2. Design G_ϵ and F_ϵ so that they represent the penalty function used in the MPC problem;

3. Solve (3.25).

The rest of this thesis assumes that this approach will give an accurate computation of the largest Lagrangian multipliers for the MPC problem. Numerical issues that can arise are discussed in Hovd and Stoican (2014).

3.5 Numerical illustrative examples

This section provides the computation of the penalty parameters to be used in the recurring illustrative examples. The previous sections described a method for guaranteeing that no constraints are violated when using soft constraints, unless the problem is infeasible. This is critical when using soft constraints to steer the system into the safety set, as mentioned in Section 1.4.2, and will become clear in Chapter 4. For now, the computation of the largest Lagrangian multipliers for the safety sets computed in Section 2.5 are provided. The computation is restricted to finding the largest $\|\lambda\|_\infty$, due to the fact that an ℓ_1-penalty function will be used.

3.5.1 Example with two states, two inputs and one dropout

Solving the optimization problem (3.25) using CPLEX[1], with the safety set (2.10) as state constraints, and the nominal input constraints (1.11), yields

$$\max_{x \in \mathbb{S}_2, u \in \mathbb{U}} \|\lambda\|_\infty = 25.14. \tag{3.26}$$

A value of $\mu = 26$ will therefore be used as the penalty parameter in order to guarantee exactness of the penalty function.

3.5.2 Example with two states, three inputs and two dropouts

Due to the inclusion of two different safety sets, i.e. one for each fault scenario, one needs to compute a penalty parameter for each set. The penalty parameter for the

[1] The code is included in the digital attachments.

first fault (actuator two dropout) is computed by the optimization problem (3.25) using the set $\mathbb{S}_2$ as state constraints the with the nominal inputs constraints. The penalty parameter for the second fault, however, needs to be computed by using $\mathbb{S}_{12}$ as the state constraint, *but with the input constraints introduced by the dropout of actuator 2.* This yields the following values:

$$\max_{x \in \mathbb{S}_2, u \in \mathbb{U}} \|\lambda\|_\infty = 61.66, \tag{3.27}$$

$$\max_{x \in \mathbb{S}_{12}, u \in \mathbb{U}_2} \|\lambda\|_\infty = 50.5235. \tag{3.28}$$

The values $\mu_2 = 62$ and $\mu_{12} = 51$ will therefore used as penalty parameters.

Chapter 4

Proactive Fault-tolerant Economic MPC

The previous chapters described the necessary theory for designing penalty functions and safety sets for the proposed proactive FTEMPC scheme. This chapter describes the scheme in detail. First, a quick introductory section including assumptions and a stability note is included. The main approach is then described before the extension of the scheme to multiple actuator faults is elaborated on and stability of the scheme is proved. The last section includes an implementation of the scheme to the recurring illustrative examples, as well as for a larger system with three states.

4.1 Introduction

The approach was briefly introduced in Chapter 1, where the different operating modes of the scheme were defined to *nominal operation*, *safe operation* and *fault operation.* These are elaborated on in Section 4.2. Several assumptions are made in the design of the scheme, which are stated the following section.

Assumptions

The scheme is designed for discrete linear time invariant systems with a convex economic cost function, and possible extensions to nonlinear systems are discussed in Chapter 6. Furthermore, the approach in this chapter assumes that no disturbances are present in the system, Chapter 5 extends the scheme to be robust in terms of disturbances.

A critical criterion for the scheme to work is that the system is controllable in the different operating modes. Therefore the following assumption is made:

Assumption 4.1. The nominal system (A, B) and the faulty system (A, B_j) are controllable, and N is chosen sufficiently large such that all *admissible* initial states

x can be steered to an admissible economic steady-state point $(x'_{\mathrm{s}}, u'_{\mathrm{s}})$ within N steps while satisfying the given state and input constraints.

Assumption 4.1 ensures that the system can be steered from any admissible initial state x to an admissible steady-state x'_{s} in N timesteps. It is important to emphasize that it is assumed that this condition holds for *any* admissible economic steady-state point, as the latter is changed by the introduction of safety constraints.

Additionally, state feedback is assumed for the system. That is, all states are perfectly measured without errors. Also, it is assumed that the FDI provides instant and accurate information about an actual fault and that either historical data or the FDI provides warning of faults that are about to occur.

A quick note on stability for economic MPC

In Chapter 1, it was noted that stability for economic MPC schemes is still a researched area, but that several contributions have been made to prove stability of the scheme. In this thesis, the stability proof of economic MPC is based on the approach in Diehl et al. (2011), where an economically optimal state is included as a terminal state in the optimization problem. The optimal steady-state x_s is computed by

$$\left\{ \min_{x,u} l\left(x,u\right) \mid x, u \in \mathbb{X} \times \mathbb{U},\ x = Ax + Bu \right\}. \tag{4.1}$$

In Diehl et al. (2011), it is shown that this guarantees stability under certain assumptions. This thesis therefore incorporates the following notation for the EMPC problem:

$$\begin{aligned}
&\min_{\mathbf{x},\mathbf{u}} && \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(4.2a)}\\
&\text{s.t.} && x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0, \dots, N-1\} && \text{(4.2b)}\\
& && x_0 = x_{\text{init}}, && && \text{(4.2c)}\\
& && x_k \in \mathbb{X}, && \forall k \in \{1, \dots, N-1\} && \text{(4.2d)}\\
& && u_k \in \mathbb{U}, && \forall k \in \{0, \dots, N-1\} && \text{(4.2e)}\\
& && x_N = x_{\mathrm{s}}, && && \text{(4.2f)}
\end{aligned}$$

where $l\left(x_k, u_k\right)$ is an economic cost function, not necessarily positive definite.

Remark 4.1. In correspondence with the control invariant terminal set described in Chapter 2, a terminal steady-state is also control invariant from the definition of a control invariant set. The steady-state is computed with constraints taken into account, and so there exists a control law satisfying the constraints that will keep the system at the steady-state once it is reached.

4.2 Approach

This section describes the main approach by providing explanations of the different operating modes, and formulates the MPC problems that are solved to compute control laws based on the current mode of the controller.

The absolute time of the system is denoted by t, while the time relative to MPC computations at the current time instant is denoted by k. The following time instants are defined:

- The controller receives information about an incipient fault at t'.
- The estimated time instant for the actual fault occurrence is denoted t_{f}.
- The fault is fixed and the actuator is brought back to normal at t_{fix}.

Figure 4.1 illustrates the approach. The system operates at a nominal steady-state point in the nominal feasible set, $\mathbb{X}_N^{\mathrm{nom}}$, until warning of an incipient fault is received, in which it is steered into the safety set and reaches a temporary steady-state which optimizes economics subject to the safety set constraint. When the fault occurs, the system is still stabilizable, due to the fact that the current state lies in the safety set. The system is steered to the economically optimal steady-state satisfying the constraints introduced by the fault.

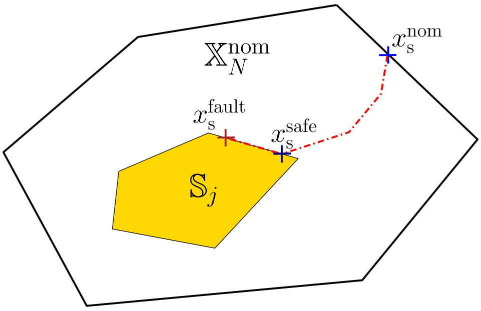

$\mathbb{X}_N^{\mathrm{nom}}$
$x_{\mathrm{s}}^{\mathrm{nom}}$
$x_{\mathrm{s}}^{\mathrm{fault}}$
$x_{\mathrm{s}}^{\mathrm{safe}}$
$\mathbb{S}_j$

Figure 4.1: Schematic illustration of the proposed scheme

4.2.1 Nominal operation

Under nominal operating conditions, the goal is to compute a feedback control for the system at hand that satisfies nominal state and input constraints. As described in the previous section, a terminal constraint will be used to guarantee stability. Given the system,

$$x_{k+1} = Ax_k + Bu_k \tag{4.3}$$

with constraints

$$x \in \mathbb{X}, \tag{4.4}$$

$$u \in \mathbb{U}, \tag{4.5}$$

the optimal steady-state $x_{\mathrm{s}}^{\mathrm{nom}}$ is given by

$$\left\{\min_{x,u} l\,(x,u) \ \mid x,u \in \mathbb{X}\times\mathbb{U},\ x = Ax+Bu\right\}. \tag{4.6}$$

The optimal feedback is thus achieved by solving at every timestep:

$$\begin{aligned}
\mathbf{P}^{\mathrm{nom}}:\ \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\,(x_k,u_k) && && \text{(4.7a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0,\ldots,N-1\} && \text{(4.7b)}\\
& x_0 = x_{\mathrm{init}}, && && \text{(4.7c)}\\
& x_k \in \mathbb{X}, && \forall k \in \{1,\ldots,N-1\} && \text{(4.7d)}\\
& u_k \in \mathbb{U}, && \forall k \in \{0,\ldots,N-1\} && \text{(4.7e)}\\
& x_N = x_{\mathrm{s}}^{\mathrm{nom}}. && && \text{(4.7f)}
\end{aligned}$$

The solution $u_0^* = u_e^{\mathrm{nom}}(x_{\mathrm{init}})$ is computed by the MPC problem (4.7) and applied to the system in a receding-horizon manner until warning of an incipient fault is received. It is clear that under nominal operating conditions, when no faults are incipient nor present, this is just the standard EMPC problem.

4.2.2 Safe operation

As soon as a warning of an incipient fault is received, from e.g. an FDI, historical data or scheduled maintenance, the system needs to be steered into a safety set where the MPC problem, when the fault occurs, is feasible. The safety set, $\mathbb{S}_j$, defined as a control invariant set when actuator j renders inactive, as described in Section 2.3, may be written as

$$\mathbb{S}_j = \{x \mid D_j x \leq d_j\}, \tag{4.8}$$

where $D_j \in \mathbb{R}^{q_s\times n}$, $d_j \in \mathbb{R}^{q_s}$, and q_s defines the number of constraints defining the polytopic safety set. The set is chosen as outlined in Chapter 2 and needs to be computed for all different combinations of actuator dropouts, however, by offline computation. Furthermore, the new temporary steady-state included in the safety set, $x_{\mathrm{s}}^{\mathrm{safe}}$ is computed by

$$\left\{\min_{x,u} l\,(x,u) \ \mid x,u \in \mathbb{X}\times\mathbb{U},\ D_j x \leq d_j,\ x = Ax+Bu\right\}. \tag{4.9}$$

The set $\mathbb{S}_j$ will often be a strict subset of $\mathbb{X}_N^{\mathrm{nom}}$, and thereby render (4.7) infeasible when operating at steady-state $x_{\mathrm{s}}^{\mathrm{nom}}$ if imposed directly as constraints in $\mathbf{P}^{\mathrm{nom}}$ at time t' for all k. Hence, the constraints (4.8) must be imposed through a penalty function, or equivalently, through soft constraints with a penalty norm. Due to the possibility of choosing a penalty parameter that makes the ℓ_1-penalty function exact, this is the choice of penalty function. The following problem is solved in order to steer the system into the safety set:

$$
\begin{aligned}
\mathbf{P}^{\text{safe}}: \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) + \mu \sum_{k=1}^{N-1} \|\epsilon_k\|_1 && && \text{(4.10a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0,\dots,N-1\} && \text{(4.10b)}\\
& x_0 = x_{\text{init}}, && && \text{(4.10c)}\\
& x_k \in \mathbb{X}, && \forall k \in \{1,\dots,N-1\} && \text{(4.10d)}\\
& u_k \in \mathbb{U}, && \forall k \in \{0,\dots,N-1\} && \text{(4.10e)}\\
& D_j x_k \leq d_j + \epsilon_k && \forall k \in \{1,\dots,N-1\} && \text{(4.10f)}\\
& \epsilon_k \geq 0 && \forall k \in \{1,\dots,N-1\} && \text{(4.10g)}\\
& x_N = x_{\text{s}}^{\text{safe}}, && && \text{(4.10h)}
\end{aligned}
$$

where the penalty parameter μ is chosen so that the penalty function is exact. The solution $u_0^* = u_e^{\text{safe}}(x_{\text{init}})$ is applied to the system in a receding-horizon manner. The soft-constraint formulation (4.10) is equivalent with optimizing a non-smooth penalty function, as described in Section 3.2, subject to the remaining constraints.

Remark 4.2. Note that the terminal constraint (4.10h) *without* the soft safety set constraint (4.10f) and the penalty function in (4.10a), is not enough by itself to drive the system into the safety set. This is due to the fact that without (4.10f) and the penalty in (4.10a), the optimal state is indeed the *nominal* steady-state $x_{\text{s}}^{\text{nom}}$ and one can therefore not guarantee that the first control action computed by the MPC problem at each timestep begins to steer the system to $x_{\text{s}}^{\text{safe}}$. The control actions that will drive the system to $x_{\text{s}}^{\text{safe}}$ are always being postponed and might then actually never be implemented. This motivates the inclusion of the soft constraint with exact penalty function in (4.10), which makes $x_{\text{s}}^{\text{safe}}$ the optimal steady-state and thus the safety set is reached before the end of the prediction horizon N.

Remark 4.3. Enforcing hard constraints $D_j x_k \leq d_j$ for $k \geq t_{\text{f}} - t'$ would not change the optimal solution $(\mathbf{x}^*, \mathbf{u}^*, \epsilon^*)$ when the penalty function is exact. If $\epsilon_k^* = 0, \forall k \geq t_{\text{f}} - t'$ is a feasible solution to $\mathbf{P}^{\text{safe}}$, then exactness of the penalty function will ensure that this indeed is the solution to $\mathbf{P}^{\text{safe}}$.

In addition to its exactness properties, the ℓ_1-penalty function is chosen for the following reason. When a fault is about to occur, it is often desirable to steer the system into the safety set as quickly as possible. This is especially true when the exact occurrence of the fault is not known in advance. For this, the following is proposed:

Proposition 4.1. *If Assumption 4.1 holds, and $\mu > \mu^*$, where μ^* is a lower threshold value to ensure that the penalty function is exact, then the solution $(\mathbf{x}^*, \mathbf{u}^*, \epsilon^*)$ to the reformulated ℓ_1 exact penalty function in $\mathbf{P}^{\text{safe}}$ will steer the state x_k inside $\mathbb{S}_j$ in the minimum number of timesteps.*

Proof. With a sufficiently large penalty parameter $\mu > \mu^*$, a feasible solution $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ to the soft constrained problem (4.10), obtained by reformulation of an exact, non-smooth penalty function, satisfies the KKT conditions of the corresponding hard-constrained problem if $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ is a feasible solution to this problem. If $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ is infeasible for (4.10), the exactness of the penalty function (4.10a) will ensure constraint satisfaction, i.e. the KKT condition, for those constraints in (4.10) that can be satisfied, i.e. the time-varying slack variables $\epsilon_k^* = 0, \forall k \geq \bar{k}$ for some $\bar{k} > 0$. Consequently, the ℓ_1 exact penalty function will yield $\epsilon_k^* > 0$ only for those k in (4.10f) that would yield infeasibility for the hard constraint $D_j x_k \leq d_j$, thereby ensuring that these constraints are violated only if necessary, and hence in the minimum number of timesteps $\bar{k}$. The assumption of convexity of the $l(x,u)$ ensures global optimality of the solution $(\mathbf{x}^*, \mathbf{u}^*, \epsilon^*)$. □

Thus, by using an exact ℓ_1-penalty function, the system is steered into the safety set in the minimum amount of timesteps. This is clearly advantageous and the most safe action to take when a fault is about to occur.

Furthermore, it is important to distinguish between two scenarios relating the estimated fault-time t_f to the prediction horizon N: If $t_\text{f} > t' + N$, then feasibility of $\mathbf{P}^\text{safe}$ at time t' will ensure $x_k \in \mathbb{S}_j$ within t_f. Else, if $t_\text{f} \leq t' + N$ then a check of ϵ^* from the solution of $\mathbf{P}^\text{safe}$ at time t' must be made. Let $\varepsilon^*_{t_\text{f}|t'}$ be the value of slack vector ε^*_k at prediction time $k = t_\text{f} - t'$ computed at sample time t'. If $\varepsilon^*_{t_\text{f}|t'} > 0$, the state cannot reach $\mathbb{S}_j$ within the estimated time t_f of actuator fault, *in which the system must be shut down or switched to an emergency mode.* Otherwise, $\varepsilon^*_{t_\text{f}|t'} = 0$, and the state is steered inside $\mathbb{S}_j$ within time t_f.

Remark 4.4. Note that the region in which the MPC is able to control the system when a certain fault is present might not necessarily change when a fault occurs. In these cases, nominal and safe operation will be the same and no proactive action is needed. However, for open-loop unstable systems that require certain actuator power to keep the system stable, it is very likely that this region will shrink. Safe operation then guarantees feasibility of the problem when the fault occurs, which would otherwise not be possible.

4.2.3 Fault operation

Assuming that the system enters the safety set before the fault occurs, the handling of the fault is the same as for a reactive scheme, i.e. that the system model and constraints are updated to represent the new system dynamics introduced by the fault. A new optimal steady-state, x_s^fault, is computed by

$$\left\{ \min_{x,u} l\left(x,u\right) \mid x,u \in \mathbb{X} \times \mathbb{U}_j,\ x = Ax + B_j u \right\}. \tag{4.11}$$

Note that adding the safety set as a constraint in problem (4.11) would not change the solution, the steady-state is computed with respect to the input constraints introduced by the fault, and will therefore implicitly lie in the feasible set for the constrained system, i.e. the safety set. The MPC problem becomes:

$$
\begin{aligned}
\mathbf{P}^{\text{fault}}: \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(4.12a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + B_j u_k, && \forall k \in \{0, \ldots, N-1\} && \text{(4.12b)}\\
& x_0 = x_{\text{init}}, && && \text{(4.12c)}\\
& x_k \in \mathbb{X}, && \forall k \in \{1, \ldots, N-1\} && \text{(4.12d)}\\
& u_k \in \mathbb{U}_j, && \forall k \in \{0, \ldots, N-1\} && \text{(4.12e)}\\
& D_j x_k \leq d_j && \forall k \in \{1, \ldots, N-1\} && \text{(4.12f)}\\
& x_N = x_{\text{s}}^{\text{fault}}, && && \text{(4.12g)}
\end{aligned}
$$

where the solution $u_0^* = u_e^{\text{fault}}(x_{\text{init}})$ is applied to the system in a receding-horizon manner. Note that the hard safety set constraint (4.12f) may be dropped due to the presence of the terminal constraint (4.12g). If (4.12f) is not present in the problem, the solution will still not yield a trajectory that leaves the safety set as this would yield infeasibility and thereby fail to satisfy the terminal constraint.

Remark 4.5. It is important to be aware of the subtle difference between an actual and an incipient fault. The MPC problem is only updated to compensate for the fault dynamics once the fault has *actually* occurred. Upon warning of the fault and before the fault occurs, the faulty dynamics has not yet been introduced to the MPC problem and the only change in the control dynamics is the addition of the soft safety set constraint.

The FTMPC scheme can be summarized as follows.

Algorithm 4.1. The following steps comprise the proactive FTEMPC scheme:

1. Offline: Compute the safety set using invariant set computation methods as described in Chapter 2.
2. Offline: Compute the largest Lagrangian multiplier for the hard safety constraint using the method presented in Chapter 3.
3. Online: In nominal operation, solve $\mathbf{P}^{\text{nom}}$ and apply the input $u_e^{\text{nom}}(x_{\text{init}})$ to the system in a receding-horizon manner.
4. Online: In safety operation, that is, when warning about an incipient fault i received, solve $\mathbf{P}^{\text{safe}}$ and apply the input $u_e^{\text{safe}}(x_{\text{init}})$ to the system in a receding-horizon manner.
5. Online: When the fault occurs, solve $\mathbf{P}^{\text{fault}}$ and apply the input $u_e^{\text{fault}}(x_{\text{init}})$ to the system in a receding-horizon manner.
6. Online: When the fault is fixed, go to step 3.

4.3 Multiple actuator faults

The previous section described the overall proposed approach for dealing with incipient faults, and focused on single actuator dropouts. This section describes how to incorporate handling of multiple actuator faults. The following notations are used. For scenarios where only single dropouts are considered, the notations are as in the previous section. That is, a warning about the fault is received at time t', and is estimated to occur at t_{f}. It will be clear from the context which actuator the fault has occurred in. Additional notations are needed for situations with multiple faults. The time of warning about an incipient fault in actuator j is denoted t'_j and the fault is estimated to occur at $t_{\text{f}j}$, for which the problems $\mathbf{P}^{\text{safe}}_{j'}$ and $\mathbf{P}^{\text{fault}}_{j}$ are solved, respectively. Additionally, the problem solved for safe operation for a current fault in actuator j, and an incipient fault in actuator i is denoted $\mathbf{P}^{\text{safe}}_{ji'}$. The problem solved for a fault in both actuator j and i is denoted $\mathbf{P}^{\text{fault}}_{ji}$.

Consider Figure 4.2, which shows the different combinations of available actuators of a system with three inputs. In the case of multiple actuator faults, safety sets and penalty parameters need to be computed for all these possible combinations. That means that one has to take each possible sequence of events into account when designing the scheme. A fault in actuator 1 followed by a fault in actuator 2, or a fault in actuator 2 followed by a fault in actuator 1, would yield the same safety set, $\mathbb{S}_{12}$. However, the two scenarios would yield different Lagrangian multipliers for the hard-constrained safety set. This is due to the fact that different actuators are used in the two scenarios for steering the system into the safety set, which yields different costs in the objective function. Thus, one needs to compute safety sets for all "nodes" in Figure 4.2, and different Lagrangian multipliers for all "vertices" in order to guarantee exactness of the penalty function in safety operation. For large systems, this might be a long procedure. However, all computations are done offline, and should therefore not be an issue in most cases.

Figure 4.3 illustrates safety sets for different combinations of faults, i.e. a fault in actuator 1, a fault in actuator 2, and a fault in both actuator 1 and 2. As illustrated, the safety sets quickly shrink when multiple faults are present. Note that the safety set $\mathbb{S}_{12}$ is a subset of both safety sets, $\mathbb{S}_1$ and $\mathbb{S}_2$. When an additional fault occurs, it is treated in the same manner as for a single dropout, but with only the remaining actuators used to steer the system into the safety set. The following sequence of events describe the approach:

1. Warning of fault in actuator 1 is received at time t'_1, the input is computed by $\mathbf{P}^{\text{safe}}_{1'}$ and the system is driven to the safety set by $\mathbb{S}_1$ using all m actuators.
2. Fault 1 occurs at time $t_{\text{f}1}$, the input is computed by $\mathbf{P}^{\text{fault}}_{1}$ and the controller is able to stabilize the system with the remaining $m-1$ actuators.
3. Warning about fault 2 at time t'_2, the input is computed by $\mathbf{P}^{\text{safe}}_{12'}$ and the system is driven to the safety set $\mathbb{S}_{12}$, using only the available $m-1$ actuators,

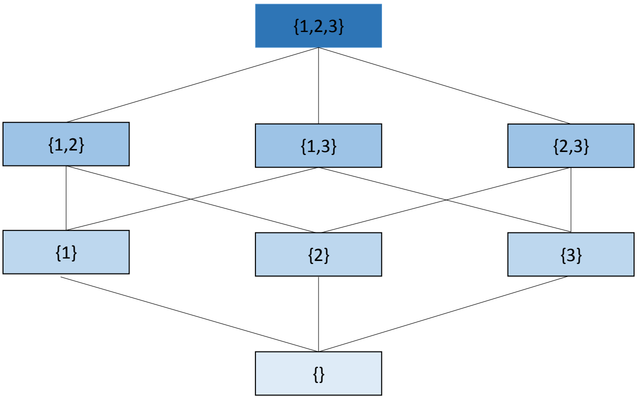


Figure 4.2: Illustration of different available actuator subsets for a system with three inputs. For example, a dropout of actuator 1 would result in the subset $\{2, 3\}$ as the available actuators.

where the system is stabilizable for a dropout of both actuator 1 and 2.

4. Fault 2 occurs at time t_{f2}, the input is computed by $\mathbf{P}_{12}^{\text{fault}}$, and the controller is able to stabilize the system with the remaining $m - 2$ actuators.

5. The procedure is repeated for additional faults. Obviously, if no actuators remain, emergency mode is necessary.

6. When faults are fixed, the system returns to the state it was in before the respective fault occurred.

Thus, one can think of additional incipient faults as a single dropout, but with the available actuators comprising the ”current” system. For example, in the case of a fault in actuator 1 and an incipient fault in actuator 2, the current system would be the system with a fault in actuator 1.

It is important to note that when linear economic cost functions are used in the MPC, the system will often operate on the boundary of the feasible set, i.e. the safety set when a fault is present. This will in many cases cause the input constraints to being active. Thus, when an additional fault is about to occur, there might not exist enough actuation power to steer the system into a new safety set, since the maximum amount of actuation power is already being spent on keeping the system on the boundary. It is therefore often advantageous to tighten the safety sets by a small amount, such that the input constraints are not active on the boundary of the safety set. The next section analyzes the stability properties of the scheme.

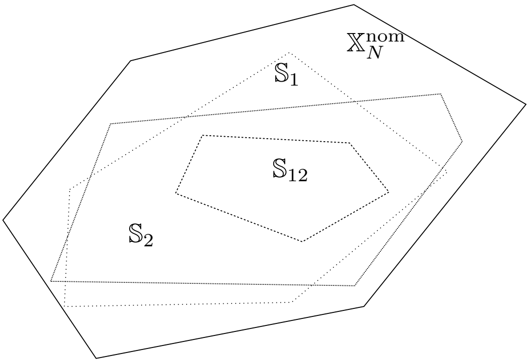


Figure 4.3: Illustration of the safety sets for faults in different actuators.

4.4 Stability

The stability proof is based on the approach in Diehl et al. (2011), and is only shown for the ℓ_1-penalty function. Additionally, it is without loss of generality based on a single actuator dropout. Although the computation of the lower bound on the penalty parameter, i.e. $\mu > ||\lambda||_{\text{D}}$, holds for any convex objective function, the following additional assumption for the stage cost $l(x,u)$ is made in order to prove stability for the scheme.

Assumption 4.2. If $l(x,u)$ contains other than linear terms, these must be strictly convex, and a constraint qualification, e.g. Slater's condition (Boyd and Vandenberghe, 2004), must additionally be satisfied at the optimal steady-state point.

If $l(x,u)$ is a linear, economic objective function, the MPC problems (4.7), (4.10) and (4.12) resort to linear programs (LPs), in which strong duality holds (Boyd and Vandenberghe, 2004, Ch. 5). The additional assumption of a constraint qualification assures strong duality to hold at optimal steady-state. If $l(x,u)$ is quadratic, R must be positive definite for quadratic terms $u^T Ru$, and a constraint qualification, e.g. Slater's condition, must in addition be satisfied for strong duality to hold.

In order to analyze stability of the proposed proactive FTEMPC scheme, three scenarios need to be considered and the "rotated" stage costs are introduced (Diehl et al., 2011),

$$L^{\text{nom}}(x,u) = l(x,u) + (x - Ax - Bu)'\lambda_{\text{s}}^{\text{nom}} - l(x_{\text{s}}^{\text{nom}}, u_{\text{s}}^{\text{nom}}), \tag{4.13a}$$

$$L^{\text{safe}}(x,u,\varepsilon) = l^{\text{safe}}(x,u,\varepsilon) + (x - Ax - Bu)'\lambda_{\text{s}}^{\text{safe}} - l(x_{\text{s}}^{\text{safe}}, u_{\text{s}}^{\text{safe}}), \tag{4.13b}$$

$$L^{\text{fault}}(x,u) = l(x,u) + (x - Ax - B_j u)'\lambda_{\text{s}}^{\text{fault}} - l(x_{\text{s}}^{\text{fault}}, u_{\text{s}}^{\text{fault}}). \tag{4.13c}$$

where

$$l^{\text{safe}}(x,u,\epsilon) \triangleq l(x_k,u_k) + \mu \sum_{i=1}^{q_s} \epsilon_{ik} \tag{4.14}$$

is the point-wise in time stage cost (4.10a) as a function of x, u and ε with ℓ_1-penalty. Moreover, $\lambda_{\mathrm{s}}^{\mathrm{nom}}, \lambda_{\mathrm{s}}^{\mathrm{safe}}$ and $\lambda_{\mathrm{s}}^{\mathrm{fault}}$ are Lagrangian multipliers for the LTI steady-state model such that strong duality holds for the three steady-state problems (4.6), (4.9) and (4.11), respectively. Note that strong duality holds by Assumption 4.2, and that by allowing slack on the constraint $H_j x \leq h_j$ only up to $N-1$, the steady-state problem of $\mathbf{P}^{\mathrm{safe}}$ is independent of ε.

Lemma 4.1. *The following relates the rotated costs* (4.13) *and the respective MPC problems:*

1. *Solving* $\mathbf{P}^{\mathrm{nom}}$ *in* (4.7) *with objective* (4.7a) *replaced with*

$$\tilde{V}_N^{\mathrm{nom}}(x_{init}) = \min \sum_{k=0}^{N-1} L^{\mathrm{nom}}(x_k, u_k)$$

gives equal solution.

2. *Solving* $\mathbf{P}^{\mathrm{safe}}$ *in* (4.10) *with the objective* (4.10a) *replaced with*

$$\tilde{V}_N^{\mathrm{safe}}(x_{init}) = \min \sum_{k=0}^{N-1} L^{\mathrm{safe}}(x_k, u_k, \epsilon_k)$$

gives equal solution.

3. *Solving* $\mathbf{P}^{\mathrm{fault}}$ *in* (4.12) *with the objective* (4.12a) *replaced with*

$$\tilde{V}_N^{\mathrm{fault}}(x_{init}) = \min \sum_{k=0}^{N-1} L^{\mathrm{fault}}(x_k, u_k)$$

gives equal solution.

Proof. All the three rotated costs are point-wise in time summed from $k = 0$ to $N-1$, and the respective MPC optimization problems have terminal equality constraint. The results hence follows immediately from Lemma 2 in Diehl et al. (2011). □

The above lemma is used directly to prove *nominal* stability of the proposed proactive FTEMPC scheme, that is, for nominal model and no disturbances.

Theorem 4.1. *(Nominal stability): If Assumption 4.1 and 4.2 hold, and* $\mu > \mu^*$ *such that the* ℓ_1*-penalty function in* (4.10) *is exact, then the following stability properties hold:*

1. *(Nominal operations):* $x_{\mathrm{s}}^{\mathrm{nom}}$ *is an asymptotically stable steady-state point of the closed-loop system* $x_{k+1} = Ax_k + Bu_e^{nom}(x_{init})$ *with Lyapunov function* $\tilde{V}_N^{\mathrm{nom}}(x_{init})$ *and region of attraction* $\mathbb{X}_N^{nom}$.

2. *(Safe operations): At time t', if (a) $t' + N \leq t_\mathrm{f}$ and $\epsilon_{t_\mathrm{f}|t'} = 0$, or (b) if $t_\mathrm{f} > t' + N$, the system will be steered inside the safety set within the t_f, in which $x_\mathrm{s}^\mathrm{safe}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + Bu_\mathrm{e}^\mathrm{safe}(x_{init})$ with Lyapunov function $\tilde{V}_N^\mathrm{safe}(x_{init})$ and region of attraction $\mathbb{X}_N^{nom}$.*

3. *(Fault operations): $x_\mathrm{s}^\mathrm{fault}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + B_j u_\mathrm{e}^{fault}(x_{init})$ with Lyapunov function $\tilde{V}_N^\mathrm{fault}(x_{init})$ and region of attraction $\mathbb{S}_j$*

Proof. A sketch of the proof is given for the three parts individually.

Part 1): Recursive feasibility of $\mathbf{P}^\mathrm{nom}$ is ensured by the terminal equality constraint $x_N = x_\mathrm{s}^\mathrm{nom}$ and Assumption 4.1. Furthermore, Assumption 4.2 ensures strong duality to hold at steady-state $x_\mathrm{s}^\mathrm{nom}$. Hence, it can be verified that $\tilde{V}_N^\mathrm{nom}(x_\mathrm{init})$ satisfies the properties of a Lyapunov function (Diehl et al., 2011, Th. 1), and in particular that

$$\tilde{V}_N^\mathrm{nom}(Ax + Bu_\mathrm{e}^\mathrm{nom}(x_\mathrm{init})) \leq \tilde{V}_N^\mathrm{nom}(x_\mathrm{init}) - L^\mathrm{nom}(x, u_\mathrm{e}^\mathrm{nom}(x_\mathrm{init})) \tag{4.15a}$$

$$\leq \tilde{V}_N^\mathrm{nom}(x_\mathrm{init}) - \beta(|x - x_\mathrm{s}^\mathrm{nom}|)) \tag{4.15b}$$

for all $x \in \mathbb{X}_N^\mathrm{nom}$, and for a K_∞-function $\beta(\cdot)$. This proves part 1) of the theorem.

Part 2): Let $0 < \bar{k} \leq t_\mathrm{f} - t'$ be an integer, such that $\varepsilon_{k|t'}^* = 0$ for all $k \geq \bar{k}$. At sample time t', let $\{\varepsilon_{0|t'}, \varepsilon_{1|t'}, \ldots, \varepsilon_{\bar{k}-1|t'}, 0, \ldots, 0\}$ be a feasible sequence of slack variables, and let $\mathbf{u}$ a feasible control sequence. Applying the feedback control law $u_\mathrm{e}^\mathrm{safe}(x_\mathrm{init})$ at time t', then at time $t'+1$, the sequence $\{\varepsilon_{1|t'}, \ldots, \varepsilon_{\bar{k}-1|t'}, 0, 0, \ldots, 0\}$ and $\{u_1, u_2, \ldots, u_{N-1}, u_\mathrm{s}^\mathrm{safe}\}$ will be feasible with $\tilde{x} = Ax + Bu_\mathrm{e}^\mathrm{safe}(x_\mathrm{init})$ as initial condition. This follows from the terminal equality constraint (4.10h) and by requiring zero slack on the constraints $H_j x \leq h_j$ at the end of the horizon. Feasibility of $\mathbf{P}^\mathrm{safe}$ for all sample times $t \geq t'$ and for all initial states $x \in \mathbb{X}_N^\mathrm{nom}$ follows by induction.

For the two scenarios of t_f relative to N, the following holds; (a) If $t' + N \leq t_\mathrm{f}$ and $\varepsilon_{t_\mathrm{f}|t'}^* = 0$, then by the recursive feasibility, exactness of the penalty term, and Proposition 4.1, the number of positive slack vectors will decrease by one for each receding-horizon iteration, decreasing the total magnitude of the ℓ_1-penalty term. Hence if $\varepsilon_{t_\mathrm{f}|t'}^* = 0$, then x_k will be steered into $\mathbb{S}_j$ within t_f, and indeed $x \in \mathbb{S}_j$ for all sample times $t \geq t_\mathrm{f}$ due to the positive invariance of $\mathbb{S}_j$. If $t_\mathrm{f} > t' + N$, then it follows immediately that $x_k \in \mathbb{S}_j$ within time t_f by feasibility of $\mathbf{P}^\mathrm{safe}$ at sample time t', and by the same arguments as above. Asymptotic stability of $x_\mathrm{s}^\mathrm{safe}$ from switching to $\mathbf{P}^\mathrm{safe}$ at time t' can then be established by using $\tilde{V}_N^\mathrm{safe}(x_\mathrm{init})$ for all $x \in \mathbb{X}_N^\mathrm{nom}$, and establishing an inequality similar to (4.15) with $L^\mathrm{safe}(x_k, u_k, \epsilon_k)$ and a K_∞-function $\tilde{\beta}(\cdot)$.

Part 3): If the MPC problem $\mathbf{P}^\mathrm{safe}$ with control law $u_\mathrm{e}^\mathrm{safe}(x_\mathrm{init})$ is able to steer the system state x_k inside $\mathbb{S}_j$ within time t_f, then for all initial states $x \in \mathbb{S}_j$, using the same arguments as in part 1) and in (Diehl et al., 2011, Th. 1), it holds that $x_\mathrm{s}^\mathrm{fault}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + B_j u_\mathrm{e}^\mathrm{fault}(x_\mathrm{init})$ with region of attraction $\mathbb{S}_j$. □

4.5 Numerical illustrative examples

This section completes the implementation of the proactive FTEMPC scheme for the illustrative examples. Additionally, an example with a system with three states is included in order to illustrate how the scheme scales for larger systems. All simulations are performed using YALMIP (Löfberg, 2004).

4.5.1 Example with two states, two inputs and one dropout

Both the safety set and the exact penalty parameter needed for this example have been computed in the previous chapters. These parameters are now used in order to fully implement the scheme for this system.

As described in the previous sections, the system operates in nominal operation where the MPC problem $\mathbf{P}^\mathrm{nom}$ is solved in a receding-horizon manner until a warning about an incipient fault is received. It is assumed that at time $t' = 20$, the warning is received from one of the methods mentioned in Chapter 1 in which the MPC controller switches to the optimization problem $\mathbf{P}^\mathrm{safe}$, in order to steer the system into the pre-computed safety set, $\mathbb{S}_2$. At time $t_\mathrm{f} = 40$, actuator 2 renders completely inactive and the MPC controller switches to the optimization problem $\mathbf{P}^\mathrm{fault}$. The fault is fixed at $t_\mathrm{fix} = 60$, and the MPC controller switches back to the original problem, $\mathbf{P}^\mathrm{nom}$.

The optimal steady-states for the different operation cases are computed by the optimization problems (4.6), (4.9), (4.11), respectively, for use as terminal constraints, as described in Chapter 4.1:

$$x_s^\mathrm{nom} = \begin{bmatrix}3.75 & 6\end{bmatrix}^T, \tag{4.16}$$

$$x_s^\mathrm{safe} = \begin{bmatrix}0.6043 & 1.6960\end{bmatrix}^T, \tag{4.17}$$

$$x_s^\mathrm{fault} = \begin{bmatrix}1.8580 & 0.7058\end{bmatrix}^T. \tag{4.18}$$

Figure 4.4 shows the system states over time, Figure 4.5 shows the input, and Figure 4.6 shows the state trajectory in the plane. The system operates in the economic optimal point, $x_\mathrm{s}^\mathrm{nom}$, until the controller receives information about the upcoming fault. The system is then driven into $\mathbb{S}_j$, and reaches the temporary steady-state point $x_\mathrm{s}^\mathrm{safe}$. Observe that it is crucial to compute $\max\limits_{x \in \mathbb{S}_2, u \in \mathbb{U}} \|\lambda\|_\infty$, since a smaller μ might not guarantee that the system is driven into the set $\mathbb{S}_j$, where the MPC controller retains feasibility when the system is affected by the fault. When the fault occurs, the system is steered to the new economic optimal steady-state point, $x_{\mathrm{s}}^{\mathrm{fault}}$. At sample time $t = t_{\mathrm{fix}}$, actuator u_2 is fixed and nominal operation is resumed, the system is driven back to its original optimal steady-state point $x_{\mathrm{s}}^{\mathrm{nom}}$.

The approach is compared with an open-loop discrete minimal-time control (DMTC) computed by the optimization problem in Appendix D. The two approaches use an equal number of timesteps to reach $\mathbb{S}_j$, while it can be seen that the minimal-time approach renders a different trajectory. In the remaining examples of this thesis, this comparison is not made. However, note that due to the theory presented in Proposition 4.1, the same result can be shown for the other examples.

With the proactive fault-tolerant MPC scheme, the system is able to retain stability after actuator 2 has a dropout, which would not be possible if the system was not steered into the safety set before the fault occurred. Nominal operation is resumed when the fault is fixed.

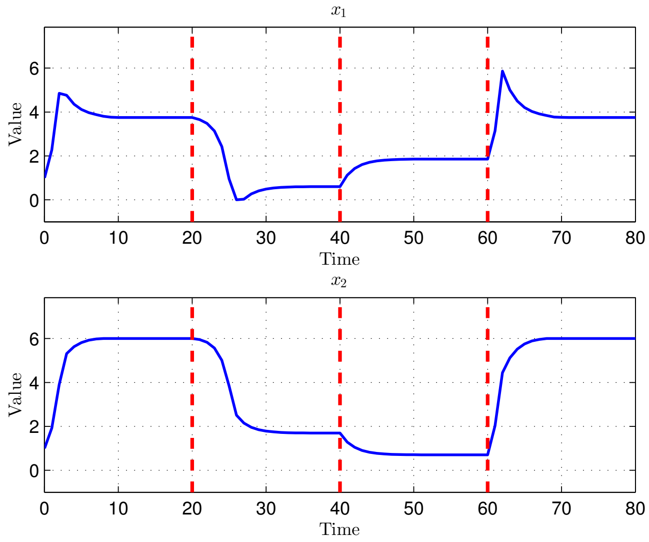


Figure 4.4: The state response of the system. The critical time instants $t' = 20$, $t_{\text{f}} = 40$, $t_{\text{fix}} = 60$ are marked by vertical lines.

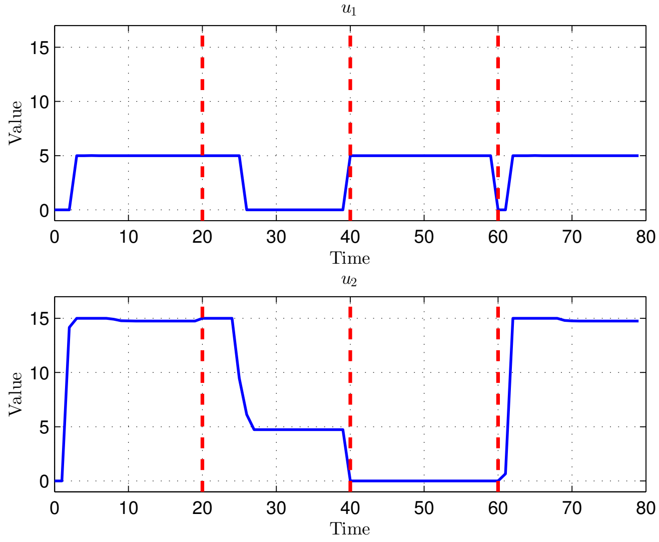


Figure 4.5: The input to the system. The critical time instants $t' = 20$, $t_{\text{f}} = 40$, $t_{\text{fix}} = 60$ are marked by vertical lines.

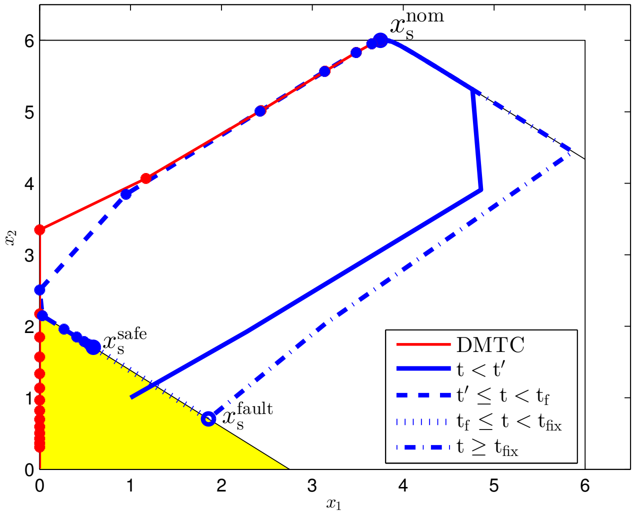


Figure 4.6: State-trajectory showing the different phases of the system evolution, as well as the optimal steady-state points. The state in each timestep is marked for both trajectories in the phase where the system is steered to the safety-set. Note that the first steps in this phase are equal for both trajectories, and that the DMTC marks (red) are difficult to notice below the blue marks.

4.5.2 Example with two states, three inputs and two dropouts

In the previous chapters, the safety sets as well as the exact penalty parameters have been computed for this example. However, as noted in Section 4.3, the safety sets have to be tightened by a small amount in order to avoid the input constraints being active on the boundary. This is done in the simulations, and as will be seen, this will cause the system to not operate on the boundaries. Recall that the scenario is focused on a situation with two actuator dropouts, the first being actuator 2, and the second being actuator 1.

As in the previous example, the optimization problem $\mathbf{P}^{\text{nom}}$ is solved in a receding-horizon manner until warning about a fault is received. The controller is warned about the first incipient fault at $t_2' = 20$, and the MPC controller switches to the optimization problem $\mathbf{P}_{2'}^{\text{safe}}$ in order to steer the system into the safety set $\mathbb{S}_2$. The fault occurs at $t_{\text{f}2} = 40$ for which the MPC problem to be solved is switched to $\mathbf{P}_2^{\text{fault}}$. At time $t_1' = 60$, the warning about a new fault in actuator 1 is received. The controller should then steer the system into the safety set $\mathbb{S}_{12}$, which is done by solving $\mathbf{P}_{2'1}^{\text{safe}}$. The fault in actuator 1 occurs at $t_{\text{f}1} = 80$ and the controller switches to solve the optimization problem $\mathbf{P}_{21}^{\text{fault}}$. The faults are fixed simultaneously and the optimization to be solved switches back to $\mathbf{P}^{\text{nom}}$ at $t_{\text{fix}} = 100$.

The following optimal steady-states are computed:

$$x_s^{\text{nom}} = \begin{bmatrix}15 & 6.1264\end{bmatrix}^T, \tag{4.19}$$

$$x_s^{\text{safe2}} = \begin{bmatrix}8.5022 & 1.6699\end{bmatrix}^T, \tag{4.20}$$

$$x_s^{\text{fault2}} = \begin{bmatrix}4.5631 & 6.0890\end{bmatrix}^T, \tag{4.21}$$

$$x_s^{\text{safe12}} = \begin{bmatrix}0.4106 & 2.9539\end{bmatrix}^T, \tag{4.22}$$

$$x_s^{\text{fault12}} = \begin{bmatrix}0.4106 & 2.9539\end{bmatrix}^T. \tag{4.23}$$

Figure 4.7 shows the system states over time, Figure 4.8 shows the input, and Figure 4.9 shows the state trajectory in the plane. Note that the input constraints are never active in their maximal value in steady-state, which is the result of the safety set tightening described in Section 4.3. The optimal steady-states in each operation are marked in Figure 4.9. Note that actuator 1 is already inactive when the fault occurs, hence $x_s^{\text{safe12}} = x_s^{\text{fault12}}$. This is not necessary for stability, but yields a larger profit for the system with this particular cost function in this example. It is clear that the system effectively steers the state into the respective safety sets, and stabilizes the system once the faults occur.

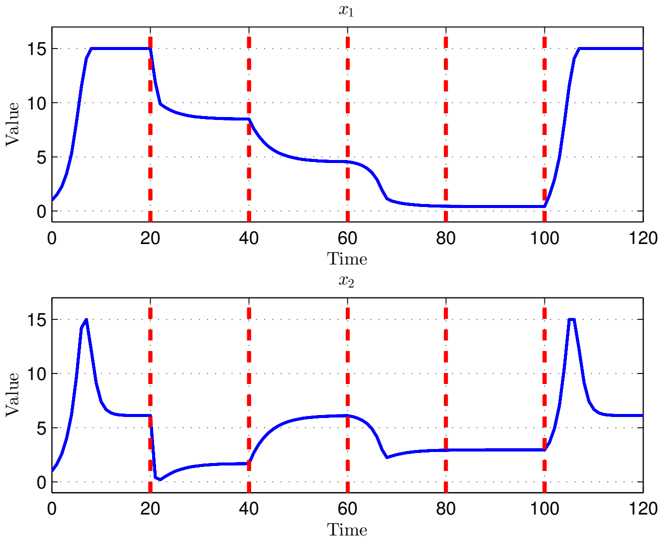


Figure 4.7: The state response of the system. The critical time instants $t_2' = 20$, $t_{\mathrm{f}2} = 40$, $t_1' = 60$, $t_{\mathrm{f}1} = 80$ and $t_{\mathrm{fix}} = 100$ are marked by vertical lines.

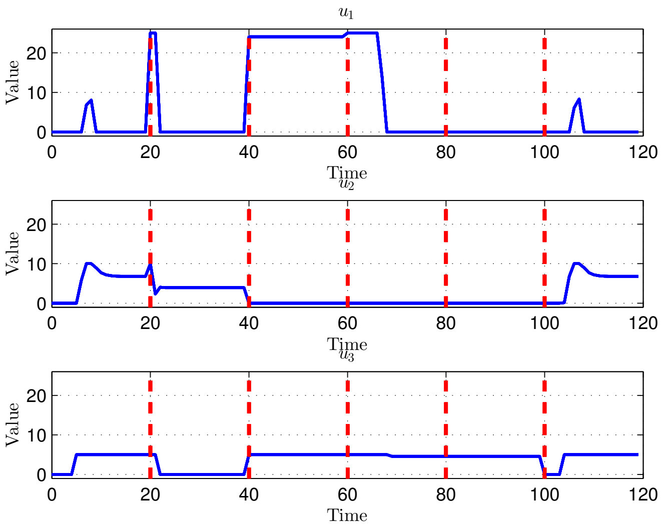


Figure 4.8: The input to the system. The critical time instants $t_2' = 20$, $t_{\mathrm{f}2} = 40$, $t_1' = 60$, $t_{\mathrm{f}1} = 80$ and $t_{\mathrm{fix}} = 100$ are marked by vertical lines.

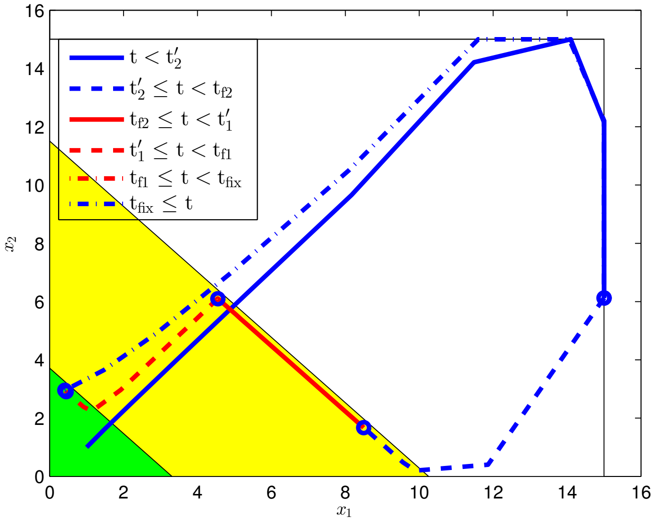


Figure 4.9: The trajectory of the system when multiple faults occur. The blue circles represent the optimal steady-states in each operation.

4.5.3 Example with three states, three inputs and one dropout

This example illustrates how the scheme scales for larger systems, and implements the scheme on a 3 × 3 system with 3 inputs.

System description

Consider again a discrete linear time invariant system

$$x_{k+1} = Ax_k + Bu_k, \tag{4.24}$$

where $x \in \mathbb{R}^3$ is the state and $u \in \mathbb{R}^3$ is the input. The system matrix is given by

$$A = \begin{bmatrix} 1.2218 & 0.0277 & 0.0218 \\ 0.0295 & 1.3340 & 0.9445 \\ 0.0183 & 0.5903 & 1.3338 \end{bmatrix}. \tag{4.25}$$

The eigenvalues of A are 2.0820, 1.2204 and 0.5871 and so the system is open-loop unstable. The input matrix in nominal operation by

$$B = \begin{bmatrix} 0.2214 & 0.0036 & 0.0046 \\ 0.0026 & 0.2572 & 0.3818 \\ 0.0015 & 0.2665 & 0.1955 \end{bmatrix}. \tag{4.26}$$

The nominal constraints on the states and inputs are given by

$$x \in \mathbb{X} = \left\{ x \mid \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leq \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix} \right\} \tag{4.27}$$

$$u \in \mathbb{U} = \left\{ u \mid \begin{bmatrix} -5 \\ -5 \\ -15 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 5 \\ 15 \end{bmatrix} \right\}. \tag{4.28}$$

The objective is to minimize the cost function

$$\sum_{k=0}^{N-1} l(x_k, u_k) = \sum_{k=0}^{N-1} (-q x_k)\,, \tag{4.29}$$

while being able to handle an incipient fault in actuator 3. The weight $q = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ is used, and a prediction horizon of $N = 5$. A shorter horizon than in the previous examples is chosen due to the fact that the computations of the penalty parameter took long to find a solution for longer horizons. As noted in Chapter 2, this is due to the fact that the safety set gets complex for higher dimensional systems. The number of constraints at each timestep are multiplied by the prediction horizon, and a longer horizon will therefore yield a larger problem to solve. This could potentially be improved by using an approximation of the safety set as noted in Chapter 2, but this is not investigated further.

Fault modeling

The situation considered is a complete dropout of actuator 3. The fault introduces new constraints on the system, given by

$$u \in \mathbb{U}_3 = \left\{ u \mid \begin{bmatrix} -5 \\ -5 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 5 \\ 0 \end{bmatrix} \right\}. \tag{4.30}$$

Safety set and penalty parameter

The safety set, $\mathbb{S}_3$, is computed as the maximal control invariant set using MPT3, as in Section 2.5. However, the set is represented as a polytope with 14 constraints, and the values are therefore not given here.

Solving the optimization problem (3.25) yields the following maximum Lagrangian multiplier

$$\max_{x \in \mathbb{S}_3, u \in \mathbb{U}} \|\lambda\|_\infty = 1.7664 \tag{4.31}$$

The penalty parameter $\mu = 2 > 1.7664$ is used in the simulations.

Implementation

It is assumed that at time $t' = 20$, a warning of an incipient fault, in which the MPC controller switches to the optimization problem $\mathbf{P}^{\text{safe}}$, in order to steer the system into the pre-computed safety set, $\mathbb{S}_3$. At time $t_{\text{f}} = 40$, actuator 2 renders completely inactive and the MPC controller switches to the optimization problem $\mathbf{P}^{\text{fault}}$. The fault is fixed at $t_{\text{fix}} = 70$, and the MPC controller switches back to the original problem, $\mathbf{P}^{\text{nom}}$.

The optimal steady-state for the different operations are

$$x_s^{\text{nom}} = \begin{bmatrix} 4 & 3 & 4 \end{bmatrix}^T \tag{4.32}$$

$$x_s^{\text{safe}} = \begin{bmatrix} 4 & 1.8 & 0.62 \end{bmatrix}^T \tag{4.33}$$

$$x_s^{\text{fault}} = \begin{bmatrix} 4 & 1.8 & 0.62 \end{bmatrix}^T . \tag{4.34}$$

When the warning of the incipient fault is received at $t' = 20$, input u_3 takes significant proactive action in order to steer the system into $\mathbb{S}_3$. Note that $x_s^{\text{safe}} = x_s^{\text{fault}}$. This is similar to the situation for the second fault in Section 4.5.2, and is simply due to the fact that u_3 is inactive at the optimal steady-state point inside the safety set before the fault occurs, even though it does not need to. Thus, the optimal steady-state does not change when it renders unusable at $t_3 = 40$. The system is driven back to its nominal optimal steady-state when the fault is fixed at $t_{\text{fix}} = 70$.

Figures 4.10 and 4.11 show that the controller is able to take proactive actions and stabilize the system once the fault occurs. This example illustrates how the scheme scales for higher dimensional systems, and provided a note that the offline computation of the penalty parameter quickly get heavy as the safety set gets more complex.

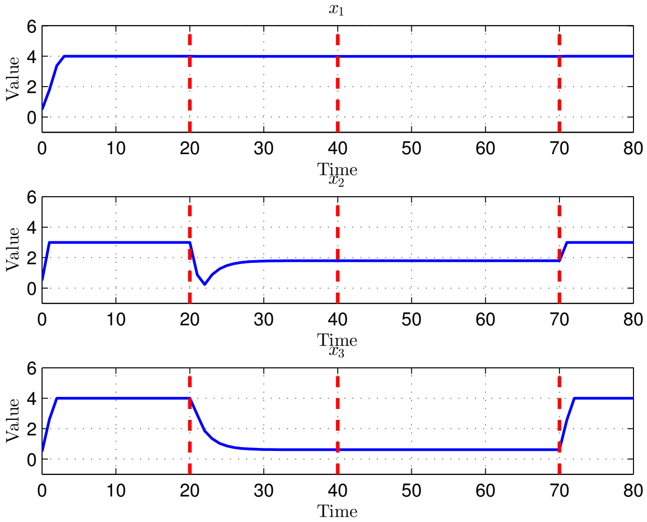


Figure 4.10: The state response of the system. The critical time instants $t' = 20$, $t_{\mathrm{f}} = 40$, $t_{\mathrm{fix}} = 70$ are marked by vertical lines.

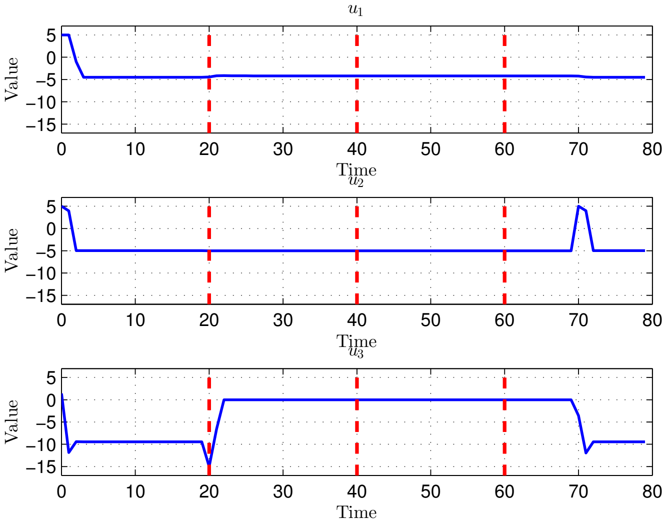


Figure 4.11: The input to the system. The critical time instants $t' = 20$, $t_{\mathrm{f}} = 40$, $t_{\mathrm{fix}} = 70$ are marked by vertical lines.

Chapter 5

Robust Proactive Fault-tolerant Economic MPC

Chapter 4 described a proactive approach for guaranteeing feasibility subject to a predicted incipient fault in an actuator. However, the approach only considered nominal systems without disturbances, which is unrealistic for many processes. This chapter describes how the scheme can be extended to be robust to unknown but bounded disturbances. First, a short review of commonly used robustness strategies for both standard tracking MPC as well as for economic MPC are described. One of these approaches is then applied together with the proactive FTEMPC scheme in order to handle disturbances in the system. The last section includes a note on stability for the proposed robust proactive fault-tolerant economic MPC scheme.

5.1 Brief review of disturbance-handling in MPC

Several methods for designing tracking MPC to be robust to disturbances are available in the literature. A frequently used approach is the *min-max* method, where the control tries to minimize the worst-case cost that could result from a future disturbance sequence by solving a min-max optimization problem, see e.g. Magni and Scattolini (2007); Raimondo et al. (2009); Lee and Yu (1997); Bemporad et al. (2003). In general, these schemes are computationally heavy, since the size of the optimization problem required grows exponentially with the increase in prediction horizon (Lee and Yu, 1997).

An approach based on constraint tightening is described in Richards and How (2006) and Marruedo et al. (2002), where the nominal constraints are tightened in a manner that guarantees that the real system with disturbances will satisfy the nominal constraints. This approach circumvents the aforementioned computational complexity, as only the nominal MPC problem with tightened constraints is solve online. However, the constraint sets will shrink drastically with the increase in prediction horizon (Yu et al., 2010).

For linear systems with additive disturbances, Mayne et al. (2005) introduce a new constraint tightening, tube-based scheme, which does not increase the computational burden, and prevents shrinking the constraint sets drastically. The scheme utilizes, in addition to the input computed by the nominal MPC, a feedback control law that aims to minimize the difference between the real system and the nominal disturbance-free system. It is shown that the state of the real system is kept within a "tube" around the trajectory of the nominal system state, hence the name tube MPC.

Robustness of *economic* MPC controllers is an active research area which recently has received increasing attention. Huang et al. (2012) presents a stability result for robust economic MPC, however, the formulation is related to tracking MPC. In Hovgaard et al. (2011), a scenario based approach is used for uncertain systems to minimize the energy consumption of a refrigeration system, while also taking probabilistic constraints into account. The authors focus on a special class of linear cost functions, and show their results and theory on different applications, but they do not provide any stability or optimality results. Another idea is presented in Muller et al. (2012), where the robustness properties at of steady-state under disturbed constraints is considered, however, no disturbances are considered within the system dynamics, only the constraints are assumed to be uncertain. In Bayer et al. (2014), the authors present a tube-based approach to achieve robustness of economic MPC subject to unknown but bounded disturbances on the states. The approach builds on the methods presented in Mayne et al. (2005) for tracking MPC, and the authors shows how it can be modified to be used within EMPC. The next section will focus on describing this latter approach, which in Section 5.3 will be applied to the proactive FTEMPC scheme to make it robust to disturbances.

5.2 Tube-based robust economic MPC

The main idea in tube MPC is to ignore the disturbance in the MPC formulation, and have the MPC computing an optimal control law and trajectory for the system without disturbances. Then, in order to counteract the effect of disturbances, an additional controller is designed to force the trajectory to lie as closely as possible to the nominal disturbance-free trajectory. It can then be shown that the real trajectory will lie in a "tube" (neighborhood) around the nominal trajectory, and thus the error is bounded (Mayne et al., 2005). In Chapter 2, positively invariant sets was shown to be a key ingredient in the design of feasible operating regions for the system. As will be seen, this is also the case in the design of tube-based control. For tube MPC, the objective is to design the controller such that the *error* lies in a positively invariant set, denoted an *invariant error set*, which is described in the next section.

First, a few notations need to be introduced. $\Phi \sim \Theta$ denotes the Pontryagin difference between the two sets Φ and Θ. $K\Omega$ denotes the multiplication of a matrix K and a set Ω, and lastly $\Phi \oplus \Theta$ denotes the Minkowski sum of the two sets Φ and Θ, see Appendix C for clarification. The notation $|z|_\Theta$ is defined to be the distance from a point z to a set Θ. The set $\mathbb{W} \in \mathbb{R}^p$ denotes the set of possible disturbance-values.

Invariant error sets

Consider the linear time-invariant disturbance-affected discrete-time system

$$\bar{x}_{k+1} = A\bar{x}_k + B\bar{u}_k + w_k, \tag{5.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times n}$, $\bar{x} \in \mathbb{R}^n$ is the system state, $\bar{u} \in \mathbb{R}^m$ is the input and $w \in \mathbb{R}^p$ is the disturbance. The bar notation is used to distinguish the system with disturbances from the disturbance-free system given by

$$x_{k+1} = Ax_k + Bu_k. \tag{5.2}$$

It is assumed that there is no model mismatch between the disturbance-free and the real system, i.e. the matrices A and B are the same in both systems. Due to the unknown disturbances in the real system dynamics (5.1), it is impossible to predict the exact system states at future time instants. However, the idea in tube MPC is to determine an invariant set of the error between the real system (5.1) and the associated disturbance-free system (5.2), which will define a bound on the error. The error is defined as

$$e_k = \bar{x}_k - x_k, \tag{5.3}$$

and hence the error dynamics are described by

$$e_{k+1} = \bar{x}_{k+1} - x_{k+1}. \tag{5.4}$$

In order to derive bounds on the error, recall the concept of robust control invariant sets defined in Definition 2.5. Based on this definition, one can make the following adaption to robust error sets.

Definition 5.1 (Robust Control Invariant Set for Error System (Bayer et al., 2014))**.** A set $\Omega \in \mathbb{R}^{n\times n}$ is robust control invariant (RCI) for the error system (5.4) if and only if there exists a feedback control law $\bar{u}_k = \phi\left(u_k, \bar{x}_k, x_k\right)$ such that for all $\bar{x}_k, x_k \in \mathbb{X}$ resulting in $e_k \in \Omega$, all $u_k \in \mathbb{U}$, and all $w_k \in \mathbb{W}$, it holds that $e_{k+1} \in \Omega$ and $\bar{u}_k \in \mathbb{U}$.

The objective is hence to design a control law such that Ω is an RCI-set (and thus bounds the error) for the error system (5.4). As described in Bayer et al. (2014), this is achieved by implementing a control law on the form

$$\bar{u}_k = \phi\left(u_k, \bar{x}_k, x_k\right) = u_e(x_{\text{init}}) + K\left(\bar{x}_k - x_k\right) \tag{5.5}$$

where $u_e(x_{\text{init}})$ is the optimal input computed by the MPC for the disturbance-free system (5.2) and the second term is an additional error feedback term in order to compensate for the disturbance. The matrix K is chosen such that $A + BK$ has all eigenvalues strictly inside the unit-circle, e.g. from linear quadratic control or pole placement, see e.g. Kalman et al. (1960).

Note that, if no disturbance is present, then $\bar{x}_k = x_k$ and the the error will always be zero, thus $\bar{u}_k = u_k$. The above procedure results in the following error dynamics

$$e_{k+1} = (A + BK)\, e_k + w_k. \tag{5.6}$$

This means that the computation of an RCI set Ω for the error system (5.4) boils down to determining an robust positively invariant (RPI) set as in Definition 2.4 for the error dynamics (5.6).

In order to find an upper bound on the error, it is necessary to compute the smallest RPI set for the error dynamics that contains all the possible values for the error. The following definition becomes useful:

Definition 5.2 (Minimal RPI (mRPI) Set (Rakovic et al., 2003))**.** The mRPI set Ω_∞ is the RPI set in $\mathbb{R}^{n\times n}$ that is contained in every closed RPI set of (5.6).

The minimal RPI set for the system (5.4) is given by (Rakovic et al., 2003)

$$\Omega_\infty \triangleq \sum_{i=0}^{\infty} (A + BK)\, E\mathbb{W} \tag{5.7}$$

i.e. the set of all possible states for the error system (5.6). This set is bounded due to the fact that $(A + BK)$ has by design only stable eigenvalues. See e.g. Rakovic et al. (2003) for a method to approximate the set Ω_∞.

Thus, by using the control law given by (5.5), the error between the real and the nominal system is bounded, and the bound can be computed by using invariant set theory.

Resulting control law

Within the framework of robust MPC, the open-loop optimization will be performed for the disturbance-free system and the sequence of inputs $\mathbf{u}$ is the optimization variable. The feedback control law (5.5), will as described, bound the error between the real system (5.1) and the disturbance-free system (5.2) to a set Ω_∞, i.e. $e_k \in \Omega_\infty$ for all $k = 0, 1, \ldots$. One can therefore guarantee that the state $\bar{x}$ of the real system will always be within a compact RCI set Ω_∞ around the state x of the disturbance-free system. However, when optimizing over the disturbance-free system, one still wants to guarantee that the real constraints are satisfied. Thus the constraints for the disturbance-free system need to be tightened accordingly, such that (Bayer et al., 2014)

$$\bar{\mathbb{X}} = \mathbb{X} \sim \Omega, \tag{5.8}$$

$$\bar{\mathbb{U}} = \mathbb{U} \sim K\Omega. \tag{5.9}$$

It can be shown that if $x_k \in \bar{\mathbb{X}}$ then $x_k \oplus \Omega_\infty \in \mathbb{X}$, and consequently $\bar{x}_k \in \mathbb{X}$. The same argument can be used to show that $\bar{u}_k \in \mathbb{U}$ (Chisci et al., 2001). Hence the nominal constrains are satisfied under the bounded disturbances. By replacing the constraints on x and u by the tightened constraints (5.8) and (5.9), the MPC problem for the system under disturbances becomes:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(5.10a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0, \ldots, N-1\} && \text{(5.10b)}\\
& x_0 = x_{\text{init}}, && && \text{(5.10c)}\\
& x_k \in \bar{\mathbb{X}}, && \forall k \in \{1, \ldots, N-1\} && \text{(5.10d)}\\
& u_k \in \bar{\mathbb{U}}, && \forall k \in \{0, \ldots, N-1\} && \text{(5.10e)}\\
& x_N = x_{\text{s}}, && && \text{(5.10f)}
\end{aligned}$$

where x_{s} is computed by the steady-state minimization

$$\left\{ \min_{x,u} l\left(x, u\right) \mid x, u \in \bar{\mathbb{X}} \times \bar{\mathbb{U}},\ x = Ax + Bu \right\}. \tag{5.11}$$

Figure 5.1 illustrates the concept of tube-based MPC. One can see that the actual trajectory lies in a compact set around the trajecory of the disturbance-free system.

Remark 5.1. In Bayer et al. (2014), the authors note that the steady-state computed by the optimization problem (5.11), might not be optimal to use as a terminal constraint when disturbances are present. In fact, a different steady-state might result in better average performance, and a new steady-state optimization problem is presented, which replaces (5.11). It is outside the scope of this thesis to investigate the average performance of the scheme, which is more of an economic MPC topic than that of fault-tolerant control. However, note that the robust proactive economic fault-tolerant control scheme presented in the next section could easily be improved by following the procedure in Bayer et al. (2014) to improve average performance.

The final control law is given by Equation (5.5). The procedure for implementing the tube-based robust economic MPC will thus be to

Algorithm 5.1 (Tube-based robust economic MPC)**.** The following steps comprise the tube-based robust EMPC scheme

1. Offline: Compute the error feedback gain K such that $A+BK$ has eigenvalues strictly inside the unit circle;

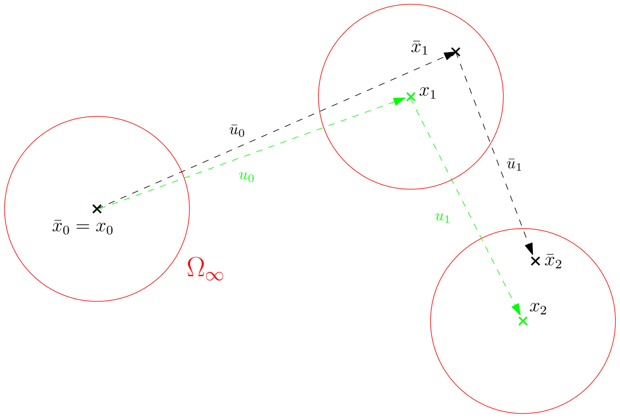


Figure 5.1: Illustration of the economic tube MPC. Adapted from Bayer et al. (2014).

2. Offline: Compute the minimal RPI set Ω_∞ for the error system (5.6);
3. Offline: Tighten the constraints according to (5.8) and (5.9);
4. Offline: Compute the optimal steady-state from (5.11);
5. Online for all $k = 0, 1, \ldots$: compute the optimal disturbance-free feedback control law $u_e(x_{\text{init}})$ from the MPC problem (5.10) and apply to system (5.2);
6. Online: for all $k = 0, 1, \ldots$: Apply $\bar{u}_k$ from Equation (5.5) to the real system.

5.3 Approach

This section describes the extension of the proactive fault-tolerant MPC scheme to be robust to additive and bounded disturbances by implementing the widely used tube MPC approach described in the previous section. This is done by using Algorithm 5.1 for the different operating modes of the proactive FTEMPC controller. The following section will describe the approach. No detailed description of the fault-tolerant approach other than the needed modifications for robustness is included, as these are given in Chapter 4.

5.3.1 Fault-free operation

For the fault-free operation case, the approach to achieve robustness to disturbances will be the same as for the standard tube economic MPC described in Section 5.2. Thus, one needs to compute the feedback gain matrix K and the minimal error robust positively invariant set, Ω_∞. The constraints on x and u are tightened according to (5.8) and (5.9), respectively. The final optimization problem to be solved at each iteration for fault-free operation is therefore given by:

$$
\begin{aligned}
\mathbf{P}_{\text{robust}}^{\text{nom}}: \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && (5.12a)\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0,\ldots,N-1\} && (5.12b)\\
& x_0 = x_{\text{init}}, && && (5.12c)\\
& x_k \in \bar{\mathbb{X}}, && \forall k \in \{1,\ldots,N-1\} && (5.12d)\\
& u_k \in \bar{\mathbb{U}}, && \forall k \in \{0,\ldots,N-1\} && (5.12e)\\
& x_N = x_{\text{s}}^{\text{nom}}, && && (5.12f)
\end{aligned}
$$

where $x_{\text{s}}^{\text{nom}}$ is computed by the steady-state minimization

$$\left\{ \min_{x,u} l\left(x,u\right) \mid x, u \in \bar{\mathbb{X}} \times \bar{\mathbb{U}},\ x = Ax + Bu \right\}. \tag{5.13}$$

The feedback law thus becomes

$$\bar{u}_k = \phi^{\text{nom}}\left(u_e^{\text{nom}}(x_{\text{init}}), \bar{x}_k, x_k\right) = u_e^{\text{nom}}(x_{\text{init}}) + K\left(\bar{x}_k - x_k\right), \tag{5.14}$$

where $u_e^{\text{nom}}(x_{\text{init}})$ is computed by the MPC problem (5.12) in a receding-horizon manner until warning about incipient fault occurs.

5.3.2 Safe operation

For safe operation, a few additional modifications need to be made. First, in addition to tightening of the nominal constraints, a new safety set $\bar{\mathbb{S}}_j$ also needs to be computed. This is done in the same manner as for the disturbance-free safety set, but with the tightened constraint sets $\bar{\mathbb{U}}_j$ and $\bar{\mathbb{X}}_j$ as inputs. These are made precise in Section 5.3.3. The safety set is then defined as the polytopic constraint

$$\bar{\mathbb{S}}_j = \left\{ x \mid \bar{D}_j x \leq \bar{d}_j \right\}. \tag{5.15}$$

The optimization problem to be solved at each iteration for safe operation is therefore given by:

$$
\begin{aligned}
\mathbf{P}_{\text{robust}}^{\text{safe}}: \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) + \mu \sum_{k=1}^{N-1} \|\epsilon_k\|_1 && && (5.16a)\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0,\ldots,N-1\} && (5.16b)\\
& x_0 = x_{\text{init}}, && && (5.16c)\\
& x_k \in \bar{\mathbb{X}}, && \forall k \in \{1,\ldots,N-1\} && (5.16d)\\
& u_k \in \bar{\mathbb{U}}, && \forall k \in \{0,\ldots,N-1\} && (5.16e)\\
& \bar{D}_j x_k \leq \bar{d}_j + \epsilon_k && \forall k \in \{1,\ldots,N-1\} && (5.16f)\\
& \epsilon_k \geq 0 && \forall k \in \{1,\ldots,N-1\} && (5.16g)\\
& x_N = x_{\text{s}}^{\text{safe}}, && && (5.16h)
\end{aligned}
$$

where the penalty parameter μ is designed to be exact, and the steady-state point $x_{\mathrm{s}}^{\mathrm{safe}}$ is computed from

$$\left\{ \min_{x,u} l\left(x,u\right) \mid x,u \in \bar{\mathbb{X}} \times \bar{\mathbb{U}},\ \bar{D}_j x \leq \bar{d}_j,\ x = Ax + Bu \right\}. \tag{5.17}$$

The feedback law hence becomes

$$\bar{u}_k = \phi^{\mathrm{safe}}\left(u_e^{\mathrm{safe}}(x_{\mathrm{init}}), \bar{x}_k, x_k\right) = u_e^{\mathrm{safe}}(x_{\mathrm{init}}) + K\left(\bar{x}_k - x_k\right), \tag{5.18}$$

where u_e^{safe} is computed from the MPC problem (5.16) in a receding-horizon manner until the fault actually occurs.

5.3.3 Fault operation

Several additional modifications need to be made in the case of fault operation. Due to the change in operating conditions, i.e. change in the input matrix B for actuator faults, a new error feedback gain matrix K_j needs to be computed. This can be done in the same manner as for K, but by using the modified input matrix B_j instead of B. Thus, K_j is designed such that $A + B_j K_j$ has its eigenvalues strictly inside the unit circle. The error dynamics in fault operation is consequently given by

$$e_{k+1} = \left(A + B_j K_j\right) e_k + w_k. \tag{5.19}$$

for which an mRPI set Ω_∞^j can be computed. Additionally, due to the change in the constraint set $\mathbb{U}$ imposed by the fault, the new constraint set $\mathbb{U}_j$ needs to be tightened according to

$$\bar{\mathbb{U}}_j = \mathbb{U}_j \sim K_j \Omega_\infty^j. \tag{5.20}$$

and the state constraints according to

$$\bar{\mathbb{X}}_j = \mathbb{X} \sim \Omega_\infty^j. \tag{5.21}$$

The disturbance-free MPC optimization problem to be solved is then given by:

$$\begin{aligned}
\mathbf{P}_{\mathrm{robust}}^{\mathrm{fault}} : \min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} l\left(x_k, u_k\right) && && \text{(5.22a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + B_j u_k, && \forall k \in \{0,\dots,N-1\} && \text{(5.22b)}\\
& x_0 = x_{\mathrm{init}}, && && \text{(5.22c)}\\
& x_k \in \bar{\mathbb{X}}_j, && \forall k \in \{1,\dots,N-1\} && \text{(5.22d)}\\
& u_k \in \bar{\mathbb{U}}_j, && \forall k \in \{0,\dots,N-1\} && \text{(5.22e)}\\
& \bar{D}_j x \leq \bar{d}_j && \forall k \in \{1,\dots,N-1\} && \text{(5.22f)}\\
& x_N = x_{\mathrm{s}}^{\mathrm{fault}}, && && \text{(5.22g)}
\end{aligned}$$

where $x_{\mathrm{s}}^{\mathrm{fault}}$ is computed by the steady-state minimization

$$\left\{ \min_{x,u} l\left(x,u\right) \mid x,u \in \bar{\mathbb{X}}_j \cap \bar{\mathbb{S}}_j \times \bar{\mathbb{U}}_j,\ x = Ax + B_j u \right\}. \tag{5.23}$$

The final feedback control law thus becomes

$$\bar{u}_k = \phi^{\text{fault}}\left(u_e^{\text{fault}}(x_{\text{init}}), \bar{x}_k, x_k\right) = u_e^{\text{fault}}(x_{\text{init}}) + K_j\left(\bar{x}_k - x_k\right). \tag{5.24}$$

where $u_e^{\text{fault}}(x_{\text{init}})$ is computed by the MPC problem (5.22) in a receding-horizon manner until the fault is fixed.

The focus has been on single actuator faults, but note that the scheme can be extended to multiple faults as described in Section 4.3, by tightening all safety sets in the manner presented in this chapter.

5.4 Stability

Section 5.3 described how the tube-based MPC approach can be used for the proposed proactive FTEMPC scheme to guarantee stability when the system is affected by additive and bounded disturbances. This section will provide a stability proof for the overall scheme.

Theorem 5.1. *(Robust stability): If Assumption 4.1 and 4.2 hold, and $\mu > \mu^*$ such that the ℓ_1-penalty function in* (5.16) *is exact, and the disturbance is bounded, then the following stability properties hold:*

1. *(Robust Nominal operations): $\mathcal{A}_{nom} = \{x_{\text{s}}^{\text{nom}}\} \times \{x_{\text{s}}^{\text{nom}}\} \oplus \Omega_\infty$ is asymptotically stable for the composite system $\bar{x}_{k+1} = A\bar{x}_k + B\phi^{nom}\left(u_e^{nom}(x_{init}), \bar{x}_k, x_k\right) + w_k$ and $x_{k+1} = Ax_k + Bu_e^{nom}(x_{init})$ with region of attraction $\bar{\mathbb{X}}_N^{nom} \times \bar{\mathbb{X}}_N^{nom} \oplus \Omega_\infty$.*
2. *(Robust Safe operations): At time t', if (a) $t' + N \leq t_{\text{f}}$ and $\epsilon_{t_{\text{f}}|t'} = 0$, or (b) if $t_{\text{f}} > t' + N$, the system will be steered inside the safety set within time t_{f}, in which $\mathcal{A}_{safe} = \left\{x_{\text{s}}^{\text{safe}}\right\} \times \left\{x_{\text{s}}^{\text{safe}}\right\} \oplus \Omega_\infty$ is asymptotically stable for the composite system $\bar{x}_{k+1} = A\bar{x}_k + B\phi^{safe}\left(u_e^{safe}(x_{init}), \bar{x}_k, x_k\right) + w_k$ and $x_{k+1} = Ax_k + Bu_{\text{e}}^{\text{safe}}(x_{init})$ with region of attraction $\bar{\mathbb{X}}_N^{nom} \times \bar{\mathbb{X}}_N^{nom} \oplus \Omega_\infty$.*
3. *(Robust Fault operations): $\mathcal{A}_{fault} = \left\{x_{\text{s}}^{\text{fault}}\right\} \times \left\{x_{\text{s}}^{\text{fault}}\right\} \oplus \Omega_\infty^j$ is asymptotically stable for the composite system $\bar{x}_{k+1} = A\bar{x}_k + B_j\phi^{fault}\left(u_e^{fault}(x_{init}), \bar{x}_k, x_k\right) + w_k$ and $x_{k+1} = Ax_k + B_j u_{\text{e}}^{\text{fault}}(x_{init})$ with region of attraction $\bar{\mathbb{S}}_j \times \bar{\mathbb{S}}_j \oplus \Omega_\infty^j$.*

Proof. Stability for the disturbance-free system can be proved in the same manner as in Chapter 4, thus the optimal steady-state points $x_{\text{s}}^{\text{nom}}$, $x_{\text{s}}^{\text{safe}}$ $x_{\text{s}}^{\text{fault}}$ are asymptotically stable for the disturbance-free system for the different plant operations. Concerning the stability of the composite system (5.1) and (5.2), the proof is based on the proof in Bayer et al. (2014), and is repeated for the three operations.

1. *(Robust Nominal operations)*: As $\bar{x}_k = x_k + e_k$ and $e_k \in \Omega_\infty$ for all $0 \leq t < t'$ and $t \geq t_{\text{fix}}$, it follows that

$$|\bar{x}_k|_{\{x_{\text{s}}^{\text{nom}}\}\oplus\Omega_\infty} \leq |x_k - x_{\text{s}}^{\text{nom}}| \leq \beta_{\text{nom}}\left(|x_0 - x_{\text{s}}^{\text{nom}}|, t\right)$$

where β_{nom} is a class $\mathcal{KL}$ function. Using this result, one obtains

$$\begin{aligned}|x_k, \bar{x}_k|_{\mathcal{A}_{\text{nom}}} =& |x_k - x_{\text{s}}^{\text{nom}}| + |x_k|_{\{x_{\text{s}}^{\text{nom}}\}\oplus\Omega_\infty} \\ \leq& 2\beta_{\text{nom}}\left(|x_0 - x_{\text{s}}^{\text{nom}}|, t\right) \leq 2\beta_{\text{nom}}\left(|x_0, \bar{x}_0|_{\mathcal{A}_{\text{nom}}}, t\right)\end{aligned}$$

for all $0 \leq t < t'$ and and $t \geq t_{\text{fix}}$. This proves part 1 of the theorem.

2. *(Robust Safe operations)* Similarly, for all $t' \leq t < t_{\text{fault}}$

$$|\bar{x}_k|_{\{x_{\text{s}}^{\text{safe}}\}\oplus\Omega_\infty} \leq \left|x_k - x_{\text{s}}^{\text{safe}}\right| \leq \beta_{\text{safe}}\left(\left|x_0 - x_{\text{s}}^{\text{safe}}\right|, t\right)$$

where β_{safe} is a class $\mathcal{KL}$ function. Using this result, one obtains

$$\begin{aligned}|x_k, \bar{x}_k|_{\mathcal{A}_{\text{safe}}} =& \left|x_k - x_{\text{s}}^{\text{safe}}\right| + |x_k|_{\{x_{\text{s}}^{\text{safe}}\}\oplus\Omega_\infty} \\ \leq& 2\beta_{\text{safe}}\left(\left|x_0 - x_{\text{s}}^{\text{safe}}\right|, t\right) \leq 2\beta_{\text{safe}}\left(|x_0, \bar{x}_0|_{\mathcal{A}_{\text{safe}}}, t\right)\end{aligned}$$

for all $t' \leq t < t_{\text{fault}}$, which proves part 2 of the theorem.

3. *(Robust Fault operations)* As $\bar{x}_k = x_k + e_k$ and $e_k \in \Omega_\infty^j$ for all $t_{\text{fault}} \leq t < t_{\text{fix}}$, it follows that

$$|\bar{x}_k|_{\{x_{\text{s}}^{\text{fault}}\}\oplus\Omega_\infty^j} \leq \left|x_k - x_{\text{s}}^{\text{fault}}\right| \leq \beta_{\text{fault}}\left(\left|x_0 - x_{\text{s}}^{\text{fault}}\right|, t\right)$$

where β_{fault} is a class $\mathcal{KL}$ function. Using this result, one obtains

$$\begin{aligned}|x_k, \bar{x}_k|_{\mathcal{A}_{\text{fault}}} =& \left|x_k - x_{\text{s}}^{\text{fault}}\right| + |x_k|_{\{x_{\text{s}}^{\text{fault}}\}\oplus\Omega_\infty^j} \\ \leq& 2\beta_{\text{fault}}\left(\left|x_0 - x_{\text{s}}^{\text{fault}}\right|, t\right) \leq 2\beta_{\text{fault}}\left(|x_0, \bar{x}_0|_{\mathcal{A}_{\text{fault}}}, t\right)\end{aligned}$$

for all $t_{\text{fault}} \leq t < t_{\text{fix}}$. This proves the last part of the theorem, and hence completes the proof.

□

5.5 Numerical illustrative example

This example illustrates the concept of using a tube MPC approach for making the proactive FTEMPC scheme robust to additive and bounded disturbances. MPT3 is used for all set-operations, and simulations are performed with YALMIP.

System description

Consider again the system in Section 1.5.1. The FTEMPC successfully stabilized the system and steered the state within a safety set. This section considers the same system, but with a change in constraints[1] and when a bounded disturbance

[1] The constraints in the aforementioned example do not contain the origin when tightened, which would mean in a practical sense that the actuators could not be turned off, and is therefore unrealistic. This example therefore considers the system where negative values are allowed in the nominal case. This will cause the tightened constraints to contain the origin.

is present. The system with disturbance will be denoted the "real" system and is given by

$$\bar{x}_{k+1} = A\bar{x}_k + B\bar{u}_k + w_k, \tag{5.25}$$

where $\bar{x} \in \mathbb{R}^2$, $\bar{u} \in \mathbb{R}^2$, $w \in \mathbb{W} \in \mathbb{R}^2$, A and B are as defined in Section 1.5.1. The disturbance set is given by

$$\mathbb{W} = \left\{ w \in \mathbb{R}^2 \ : \ \begin{bmatrix} -0.12 \\ -0.12 \end{bmatrix} \leq \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \leq \begin{bmatrix} 0.12 \\ 0.12 \end{bmatrix} \right\}. \tag{5.26}$$

The constraints are given by

$$x \in \mathbb{X} = \left\{ x \in \mathbb{R}^2 \mid \begin{bmatrix} -6 \\ -6 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 6 \\ 6 \end{bmatrix} \right\}, \tag{5.27}$$

$$u \in \mathbb{U} = \left\{ u \in \mathbb{R}^2 \mid \begin{bmatrix} -5 \\ -15 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 15 \end{bmatrix} \right\}. \tag{5.28}$$

The objective is to minimize the cost function

$$\sum_{k=0}^{N-1} l(x_k, u_k) = \sum_{k=0}^{N-1} \left(-qx_k\right), \tag{5.29}$$

while being able to handle an incipient fault in actuator 2 and attenuating disturbances. The weight $q = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ is used. The disturbance is implemented as a random vector with values between $-0.12 \leq w \leq 0.12$ at each timestep.

Fault modeling

The same time instants for the fault warning, fault occurrence and fault fix are used, i.e. the warning is received at t', $u_2 = 0$ for $t_\mathrm{f} \leq t < t_\mathrm{fix}$ where $t' = 20$, $t_\mathrm{f} = 40$ and $t_\mathrm{fix} = 60$. The new input constraints due to the presence of the fault are given by

$$u \in \mathbb{U}_2 = \left\{ u \in \mathbb{R}^2 \mid \begin{bmatrix} -5 \\ 0 \end{bmatrix} \leq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 0 \end{bmatrix} \right\}. \tag{5.30}$$

Error feedback gains and mRPI computations

The error feedback gains K and K_2 are computed by linear quadratic control using the `dlqr` command in MATLAB with A as the system matrix, B and B_2 as the respective input matrices and with identity weights on both states and inputs. This results in:

$$K = \begin{bmatrix} 1.2706 & 1.6836 \\ 1.4200 & 1.6805 \end{bmatrix}, \tag{5.31}$$

$$K_2 = \begin{bmatrix} 2.7445 & 3.4417 \\ 0 & 0 \end{bmatrix}. \tag{5.32}$$

The mRPI sets for the error system, Ω_∞ and Ω_∞^2, are computed from the toolbox described in Riverso et al. (2013). These sets are represented as polytopes with many constraints and are therefore not given here.

Constraint tightening

As described throughout the chapter, it is necessary to tighten the constraints. Following the procedure outlined in Section 5.3, yields:

$$\bar{\mathbb{X}} = \mathbb{X} \sim \Omega_\infty = \left\{ x \ : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 5.7057 \\ 5.7121 \\ 5.7057 \\ 5.7121 \end{bmatrix} \right\}, \tag{5.33}$$

$$\bar{\mathbb{X}}_2 = \mathbb{X} \sim \Omega^2_\infty = \left\{ x \ : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 5.6582 \\ 5.7239 \\ 5.6582 \\ 5.7239 \end{bmatrix} \right\}, \tag{5.34}$$

$$\bar{\mathbb{U}} = \mathbb{U} \sim K\Omega_\infty = \left\{ u \ : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} u \leq \begin{bmatrix} 4.463 \\ 14.44 \\ 4.463 \\ 14.44 \end{bmatrix} \right\}, \tag{5.35}$$

$$\bar{\mathbb{U}}_2 = \mathbb{U}_2 \sim K_2\Omega^2_\infty = \left\{ u \ : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} u \leq \begin{bmatrix} 3.865 \\ 0 \\ 3.865 \\ 0 \end{bmatrix} \right\}. \tag{5.36}$$

Safety set and penalty parameter

The nominal safety set $\mathbb{S}_2$ is computed using the nominal constraints as inputs. It is only computed for illustrative purposes, and is not used in the simulations. The tightened safety set $\bar{\mathbb{S}}_2$ is computed using MPT3 with the tightened constraints $\bar{\mathbb{X}}_2$ and $\bar{\mathbb{U}}_2$ as inputs, which yields

$$\bar{\mathbb{S}}_2 = \left\{ x \ : \begin{bmatrix} 0.6202 & 0.7845 \\ -0.6202 & -0.7845 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 1.3188 \\ 1.3188 \\ 5.6582 \\ 5.7239 \\ 5.6582 \\ 5.7239 \end{bmatrix} \right\}. \tag{5.37}$$

The nominal safety set, as well as the tightened safety set is included in Figure 5.2. Due to the tightening of the safety set and change in objective function, a new penalty parameter needs to be computed. Solving the optimization problem (3.25) with the aforementioned constraints, yields

$$\max_{x \in \bar{\mathbb{S}}_2, u \in \bar{\mathbb{U}}} \|\lambda\|_\infty = 26.92. \tag{5.38}$$

A penalty value of $\mu = 27 > 26.92$ is used in the simulation.

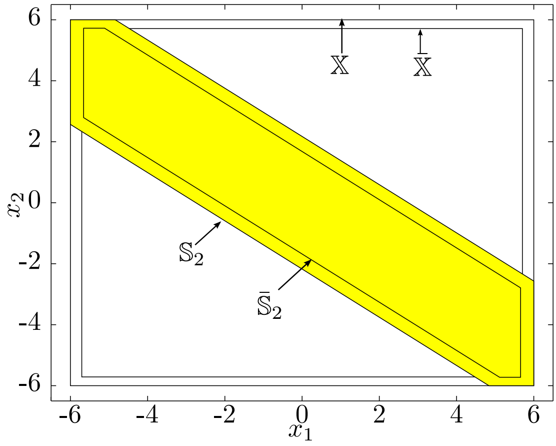


Figure 5.2: Illustration of the nominal and tightened safety sets, as well as the nominal and tightened state constraint sets.

Steady-state computations

The following optimal steady-states are computed for the different operations, which are used as terminal constraints

$$x_s^{\text{nom}} = \begin{bmatrix}3.442 & 5.712\end{bmatrix}^T , \tag{5.39}$$

$$x_s^{\text{safe}} = \begin{bmatrix}1.5833 & 0.4166\end{bmatrix}^T , \tag{5.40}$$

$$x_s^{\text{fault}} = \begin{bmatrix}1.4370 & 0.5451\end{bmatrix}^T . \tag{5.41}$$

Implementation

The controller operates in nominal operation where the MPC problem $\mathbf{P}_{\text{robust}}^{\text{nom}}$ is solved in a receding-horizon manner, and the control law (5.14) is applied to the system until a warning about an incipient fault is received. At time $t' = 20$, a warning is received of an incipient fault in actuator 2 in which the MPC controller switches to the optimization problem $\mathbf{P}_{\text{robust}}^{\text{safe}}$ and applies the control law (5.18), in order to steer the system into the pre-computed, tightened safety set, $\bar{\mathbb{S}}_2$. At time $t_{\text{f}} = 40$, actuator 2 renders completely inactive and the MPC controller switches to the optimization problem $\mathbf{P}_{\text{robust}}^{\text{fault}}$, and the control law (5.24) is applied to the system. The fault is fixed at $t_{\text{fix}} = 60$, and the MPC controller switches back to the original problem, $\mathbf{P}_{\text{robust}}^{\text{nom}}$.

Figures 5.3 and 5.4 show the state evolution and the input, respectively. Figure 5.5 shows the trajectory of the system, and Figure 5.6 shows the error in the plane between the real and the disturbance-free system, i.e. $e_1 = \bar{x}_1 - x_1$ and $e_2 = \bar{x}_2 - x_2$. It is clear that the error never exceeds the pre-computed bounds, which corresponds to the result in Figure 5.3, where the real system is kept within a tube around the disturbance-free system. The tightening of the input constraints also results in the nominal input constraints never being violated, as seen in Figure 5.4. However, note also that the error bound in Figure 5.6 might be conservative as the error states never reach the boundary.

As in the disturbance-free example in Section 4.5.1, the controller is able to steer the system inside a safety set upon warning of the fault, and stabilize the system once the fault occurs.

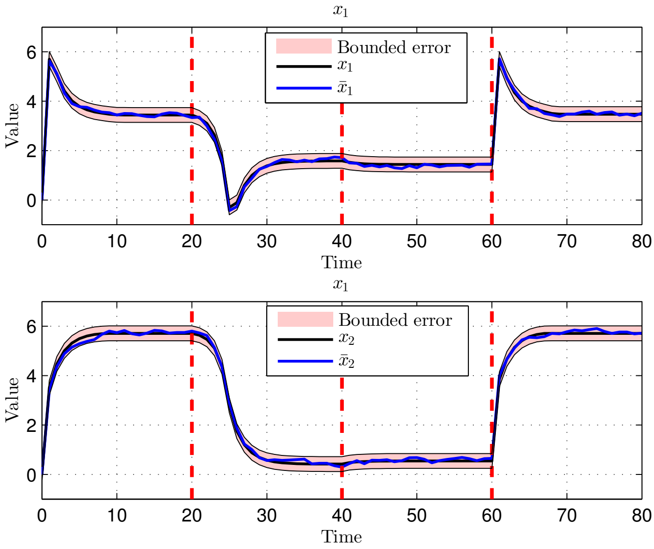


Figure 5.3: The blue line shows the state evolution of the real system, $\bar{x}$, while the black line shows the state evolution of the disturbance-free system, x. Notice that the real system evolution lies in a tube around the disturbance-free response.

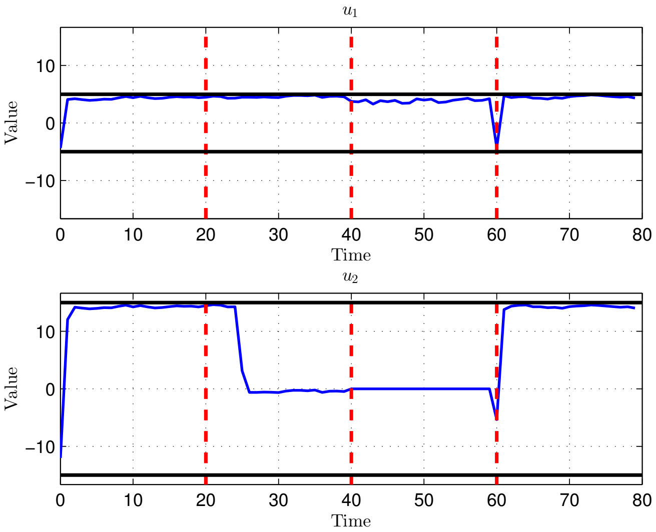


Figure 5.4: The input to the system. The horizontal black lines symbolize the nominal input constraints. It is clear that the nominal input constraints are satisfied due to the tightening of constraints.

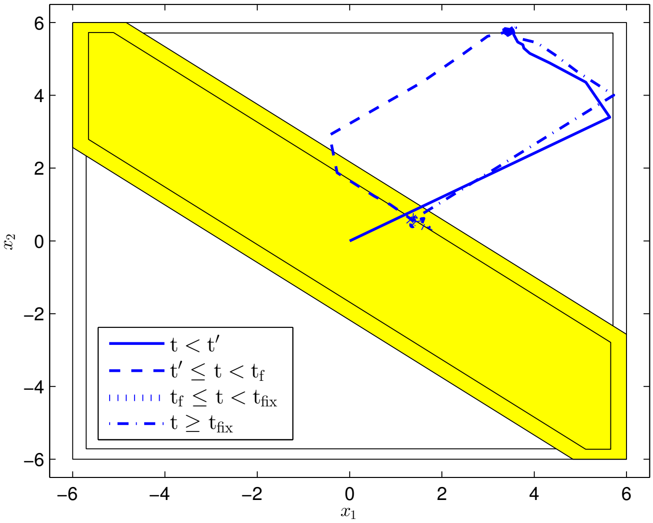
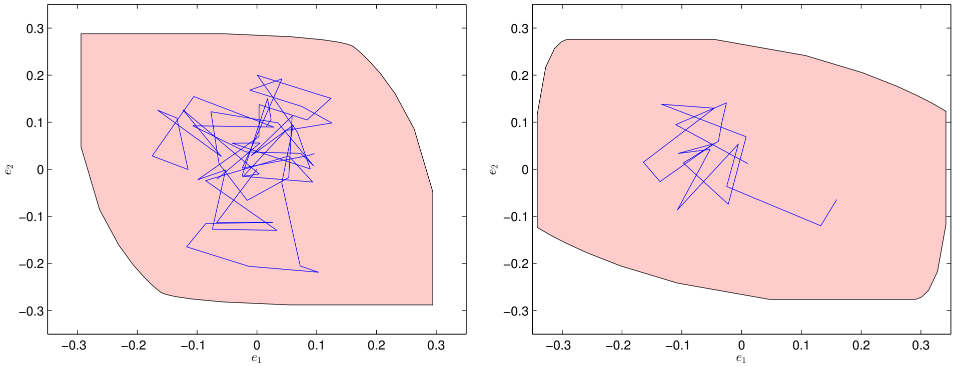


Figure 5.5: The system trajectory. The inner yellow region represents the tightened safety set.



(a) Error trajectory in nominal and safe operation.

(b) Error trajectory in fault operation.

Figure 5.6: The trajectory of the error between the real and the disturbance-free system and the respective invariant sets for the error systems. The response is separated into two different plots, due to the different values of the error bound (Ω_∞ and Ω^2_∞) for the different operations.

Chapter 6

Discussion

This thesis describes a proactive fault-tolerant economic MPC approach, including stability proofs and numerical examples to illustrate the scheme. This chapter includes a discussion of the results, and the assumptions made for the development of the FTEMPC scheme.

6.1 On the assumptions for the approach

As noted in Section 4.1, several assumptions are made in order for the scheme to work. A critical assumption is that the system is assumed to be controllable for all the operations of the controller. However, the assumptions of controllability would be needed for any type of controller, and is therefore not discussed further since it does not contribute to the discussion of the scheme.

State feedback is assumed throughout the thesis, which is often not available for real systems. In the case of output feedback where some states would need to be estimated, and uncertainty in the measurements are present, the scheme would need to be adapted. Due to the fact that linear economic objective functions are used, the optimal steady-state will often lie on the boundary of the safety set in safe and fault operations. Thus, uncertainty in the exact value of the states could potentially lead to a response where the system is perturbed outside the safety set in fault operation, and an infeasible MPC problem would occur. This is clearly not a desired situation, since this would lead to instability for many systems. Additional tightening of the safety set might therefore be needed to ensure that the system does not exit the set due to model uncertainty. This is a matter of making the scheme robust to noise on the measurements and uncertainty in the model, which is an important extension.

Furthermore, it is assumed that information about incipient faults are available. Clearly, one can not always make sure that this holds. However, for slowly degrading performance in actuators, as well as for scheduled maintenance where the actuator is taken out of action at a known time instant, one can often make sure that this information is available. Additionally, as noted, several history-based approaches are available for this type of estimation. In the case where sudden complete dropouts of actuators without warning is possible, the system should always be operating in the safety set to ensure stability when the fault occurs. Note that in this case, a reactive reconfiguration of the controller would still be necessary, since the proactive scheme only prepares the system for a fault, and is not sufficient by itself for fault-tolerant control.

Another critical assumption is the instant detection and accurate estimation of an actual fault, which makes it possible for instant reconfiguration of the MPC problem in order to handle the fault. Realistically, fault detection schemes will use a certain amount of timesteps before the fault is detected. Thus, there is a critical time window after the fault occurs until it is detected. Since there will be a severe model mismatch between the MPC and the real system in this time period, there is a chance that the system will be steered outside the safety set and become unstable. This is due to the fact that, even though the system operates in a control invariant set, the MPC does not compute the control law to keep the system inside this set. This further motivates the fact that reactive reconfiguration is necessary in addition to the proactive part of the FTEMPC. The scheme therefore needs to be adapted to handle this issue. However, the time it takes for an FDI unit to detect a fault is often correlated with its severity, and since the scheme mostly deals with complete dropouts, the assumption of quick detection is often valid.

The scheme is designed for linear systems. In the case of nonlinear systems, several extensions and adaptations are needed. The computation method of the penalty parameter described in Chapter 3, will no longer yield a correct lower bound, since it relies on a convex MPC problem for which the KKT-conditions guarantees a unique global optimal solution. A nonlinear system will yield a non-convex MPC problem, and the approach can no longer be used to find the global maximum Langrangian multiplier. Additionally, the computation of control invariant sets were mentioned in Chapter 2 to be easily implemented for linear system. For nonlinear systems, however, this is not the case. The scheme therefore relies on the development of computational methods for control invariant sets for nonlinear systems.

A possible alternative to the computation of control invariant safety sets, is to directly penalize the input corresponding to the actuator with an incipient fault, using an exact penalty function. In order to yield a feasible solution while still minimizing the use of the penalized input, the MPC should compute a trajectory which implicitly enters the safety set. This can be thought of as a ”soft” shut-down of the actuator, thereby circumventing the feasibility issues of abruptly rendering it unusable, and should yield a similar trajectory as when steering the system in to a pre-computed safety set. This might be a better approach in the case of nonlinear systems, and is mentioned as potential future work in Chapter 7.

6.2 Implementing the scheme and numerical results

This section discusses the implementation of the scheme and considerations relating maximizing profit and safe operation of the system.

6.2.1 Without disturbances

The thesis contains two recurring illustrative examples, illustrating single and multiple actuator dropouts, as well as two additional examples to illustrate how the scheme scales for larger systems, and how the scheme is extended to handle disturbances. The examples illustrate that the system is steered into a safety set using the designed exact penalty function, and that the controller is able to stabilize the system once the fault occurs. The approach outlined in Diehl et al. (2011) for ensuring economic MPC stability was implemented together with the scheme. Note that the inclusion of a terminal constraint is a conservative criteria, making the feasible set in all operations smaller. However, EMPC has received increasing attention in the last years, and as the development of EMPC continues, this criteria might be possible to relax. An approach for relaxing this criteria is described in Angeli et al. (2012).

Furthermore, the example in Section 4.5.1 compares the trajectory of the exact penalty approach for steering the system into the safety set to a minimum time approach. The two approaches use the same amount of timesteps to reach the safety set, however with different trajectories. This confirms Proposition 4.1, that the choice of an exact ℓ_1-penalty function results in the minimum amount of timesteps to reach the safety set during safe operation of the controller. This is clearly advantageous in situations where the exact time instant of the fault is not available, and only an estimate of the time of occurrence is known. On the other hand, if the exact time instant for the fault occurrence for the fault was known in advance, one could possibly compute the number of timesteps needed to reach the safety set, and have the system just reach the set in time before the fault occurs. This would let the system operate at the nominal economic optimal state for a longer time period, and hence yield larger economic profits. However, there is a clear trade-off between making sure the safety set is reached before the fault occurs, and delaying the steering process. The idea of delaying the safety process of steering the system to the safety set is generally non robust in terms of small model mismatches or uncertainty in the estimate of the fault occurrence, as this could hamper the system in reaching the safety set in time. Thus, for realistic implementations, the safest approach would be to reach the safety set as quickly as possible.

Additionally, it is important to consider how far away from the safety set the system is allowed to operate in nominal operation. The further away from the safety set the system operates, the longer it will take for it to be reached. Thus, if the time window between the warning of the fault and the occurrence of the fault is small, the system might not reach the safety set before the fault occurs if it operates in a nominal state that is far away from the safety set. Again, there will always be a trade-off between safety and profitability, and this is something that should be taken in to account if the scheme were to be implemented on a real system. The larger the aforementioned time window is, the further away from the set the system can operate while still making sure it will be reached before the fault occurrence.

The example in Section 4.5.2 illustrates the approach for multiple faults, and the simulations confirm the theory from Section 4.3. By designing safety sets and penalty functions for each possible scenario, the scheme is effectively extended to handle multiple faults. This was expected, as the extension is simply a repetition of the implementation process for each possible fault scenario. However, it was noted that the safety sets needed to be reduced by a small amount to avoid the input constraints being active. This is due to the fact that active inputs are at their maximum values, and when warning of a new incipient fault is received, there might not be enough remaining actuation to steer the system into the new safety set. It is therefore critical to tighten the safety set in cases where multiple faults are possible.

In Section 4.5.3, the scheme was implemented on a larger, 3×3 system. The representation of the safety set was observed to quickly get more complex as the number of states in the system increased, making the number of constraints needed to represent the set larger. This severely increases the offline computation time of the Lagrangian multipliers to compute the penalty parameter μ, as the increase in constraints are multiplied by the length of the horizon. This motivates the development of approximation methods for invariant sets that represent the sets by fewer constraints, as noted in Section 2.4. The scheme was effectively applied to the larger system once the offline computations were made. Thus, the main challenge of implementing the scheme for larger systems is the efficiency of the offline computations for μ to be exact. With the current rapid increase in computational power and the developments of algorithms for these computations, their accuracy might be a reasonable assumption.

Section 4.2.3 also notes that the feasible region might not change when an actuator drops out. The system will then already be operating in the safety set, and no additional action is needed. However, it is important to note that the feasible set *might* change, and failure to prepare the system for a fault may result in an unstable system response.

6.2.2 With disturbances

In Chapter 5, the scheme is extended to handle unknown but bounded additive disturbances on the states. The widely used tube MPC approach is implemented to make the scheme robust to these disturbances. The constraints are tightened according to a computed error bound and an additional term in the feedback law is included that aims to minimize the error between the real and the disturbance-free system. The example in Section 5.5 illustrates that the extended scheme effectively attenuates the unknown disturbances by forcing the system to lie in a tube around the disturbance-free system. It is also observed that the error never exceeds the computed error bound, which confirms the theory on error invariant sets presented in Chapter 5. Thus, all properties such as the minimum time result using the exact ℓ_1-penalty function still holds when implementing the scheme on disturbance affected systems. This clearly motivates the choice of using tube MPC in contrast to other disturbance attenuation schemes together with the proactive FTEMPC scheme.

However, the estimation of the error bound might be conservative due to the fact the bound is never reached, as can be seen in Figure 5.6. Furthermore, the more conservative the bound, the more the constraints are tightened, making the feasible regions for both nominal, safe, and fault operation smaller. It is outside the scope of this thesis to further investigate tube MPC as an isolated scheme, but it is noted that the accuracy of the error bound estimation is critical for ensuring robustness, while not tightening the constraints more than necessary. Additionally, it is economically desirable that the scheme can be improved by following the method in Bayer et al. (2014) for computing more economically optimal steady-state points for systems affected by disturbances.

The scheme relies on theory from several research areas, i.e. designing exact penalty functions, invariant sets, and theory from economic MPC. Different challenges in all areas exist. However, several references were given for new and prospering theory for handling these challenges, and assuming these areas continue to grow at the current rate, the scheme shows promising results. Not many proactive schemes exist in the literature, expect for the Lyapunov MPC in Lao et al. (2013) and the hybrid scheme of Bø and Johansen (2014). Thus, the proposed proactive FTEMPC approach differs from most schemes in the literature in that it takes proactive action to prepare the system for a fault rather than only relying on reactive fault-tolerance.

Chapter 7

Conclusion

This thesis describes a proactive fault-tolerant economic model predictive control approach for handling incipient actuator faults, while still maintaining economic operation of the system. The theory required for developing the proposed scheme, and the key ingredients in the approach are presented in their respective chapters. Two recurring examples are used throughout the thesis that illustrate the theory as it is presented, before these examples are implemented with the final scheme. Additional examples were included in order to show how the approach scales for larger systems, as well as to illustrate the extension to incorporating disturbance attenuation in the approach. The numerical examples show that the approach is an effective way of dealing with incipient actuator dropouts, and that by using soft constraints and exact penalty functions, the system is effectively steered into a pre-computed safety set where the controller is able to stabilize the system once the fault occurs. Stability was proven under certain assumptions, both for systems with and without disturbances.

Furthermore, it was noted that the main challenge of implementing the proposed scheme on larger systems was the offline computations for computing a lower bound on the penalty parameter. This was due to the fact that the number of constraints in the problem increases with the number of states and inputs, which severely increases the problem size. Additionally, invariant sets for representing the safety set quickly become complex for larger systems, which also contributes to a significant increase in the number of constraints. Methods for approximating these sets should therefore be implemented for larger systems, in order to reduce computational load, as well as for increasing numerical accuracy.

As discussed in Chapter 6 the scheme relies on theory from several different research areas, i.e. theory for designing exact penalty functions, invariant sets and from economic MPC. Assuming that these research fields continue to expand, the scheme shows promising results.

Chapter 8

Future work

The assumptions made for the development of the proposed proactive fault-tolerant economic MPC scheme might not be realistic for all systems. Several extensions and adaptations are needed when these assumptions are no longer valid. The assumptions were discussed in Section 6.1, and future work therefore includes extending the scheme to relax these assumptions. Additionally, the computations for designing the penalty parameter became demanding for larger systems. This was especially true when the representation of the control invariant safety set became complex. Future work therefore includes implementing methods for simpler approximations of these sets.

Furthermore, extending the approach to work for nonlinear systems requires significant additional work. As mentioned in Chapter 6, this will require a new method for computing maximum Lagrangian multipliers, as the approach described in Chapter 3 is restricted to linear systems. Additionally, methods for computing control invariant sets for nonlinear systems, or an approach that directly penalizes the input corresponding to the actuator with an incipient fault should be developed.

It was also noted that it is economically desirable to compute new steady-state values when the scheme is used on systems with disturbances. Thus, future work also includes implementing the approach outlined in Bayer et al. (2014) for computing new optimal steady-states.

Bibliography

Amrit, R., Rawlings, J. B., and Angeli, D. (2011). Economic optimization using model predictive control with a terminal cost. *Annual Reviews in Control*, 35(2):178–186.

Anderson, D. and Osborne, M. (1976). Discrete, linear approximation problems in polyhedral norms. *Numerische Mathematik*, 26(2):179–189.

Angeli, D., Amrit, R., and Rawlings, J. B. (2012). On average performance and stability of economic model predictive control. *IEEE Transactions on Automatic Control*, 57(7):1615–1626.

Athanasopoulos, N. and Bitsoris, G. (2010). Invariant set computation for constrained uncertain discrete-time linear systems. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 5227–5232. IEEE.

Bayer, F. A., Müller, M. A., and Allgöwer, F. (2014). Tube-based robust economic model predictive control. *Journal of Process Control*, 24(8):1237–1246.

Bemporad, A., Borrelli, F., and Morari, M. (2003). Min-max control of constrained uncertain discrete-time linear systems. *Automatic Control, IEEE*, 48(9):1600–1606.

Blanchini, F. (1994). Ultimate boundedness control for uncertain discrete-time systems via set-induced lyapunov functions. *IEEE Transactions on Automatic Control*, 39(2):428–433.

Blanchini, F. (1995). Nonquadratic Lyapunov functions for robust control. *Automatica*, 31(3):451–461.

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

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization.* Cambridge University Press, New York.

Bø, T. I. and Johansen, T. A. (2014). Dynamic Safety Constraints by Scenario Based Economic Model Predictive Control. In *IFAC World congress*, number 2009.

Chisci, L., Rossiter, J. A., and Zappa, G. (2001). Systems with persistent disturbances: predictive control with restricted constraints. *Automatica*, 37(7):1019–1028.

Colson, B., Marcotte, P., and Savard, G. (2005). Bilevel programming: A survey. *4OR*, 3(2):87–107.

Demetriou, M. A. and Polycarpou, M. P. (1998). Incipient fault diagnosis of dynamical systems using online approximators. *IEEE Transactions on Automatic Control*, 43(11):1612–1617.

Diehl, M., Amrit, R., and Rawlings, J. B. (2011). A Lyapunov function for economic optimizing model predictive control. *IEEE Transactions on Automatic Control*, 56(3):703–707.

Ellis, M. and Christofides, P. D. (2014). Economic Model Predictive Control with Time-Varying Objective Function for Nonlinear Process Systems. *AIChE Journal*, 60(2):507–519.

Fletcher, R. (1987). *Practical Methods of Optimization.* John Wiley & Sons, second edition.

Fortuny-Amat, J. and McCarl, B. (1981). A representation and economic interpretation of a two-level programming problem. *Journal of the operational Research Society*, pages 783–792.

Fukuda, K. (2004). Frequently asked questions in polyhedral computation. *Technical report* [Online]. `http://www.ifor.math.ethz.ch/~fukuda/polyfaq/polyfaq.html`.

Hovd, M. (2011). Multi-level programming for designing penalty functions for mpc controllers. In *Proceedings of the 18th IFAC World Congress*, volume 18, pages 6098–6103.

Hovd, M. and Braatz, R. D. (2001). Handling state and output constraints in mpc using time-dependent weights. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 2418–2423. IEEE.

Hovd, M. and Stoican, F. (2014). On the design of exact penalty functions for MPC using mixed integer programming. *Computers & Chemical Engineering*, 70(5):104–113.

Hovgaard, T. G., Larsen, L. F., and Jørgensen, J. B. (2011). Robust economic mpc for a power management scenario with uncertainties. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 1515–1520. IEEE.

Huang, R., Biegler, L. T., and Harinath, E. (2012). Robust stability of economically oriented infinite horizon nmpc that include cyclic processes. *Journal of Process Control*, 22(1):51–59.

Isermann, R. and Ballé, P. (1997). Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 5:709–719.

Janesch, S. and Santos, L. (1997). Exact penalty methods with constrained subproblems. *Investigacióon Operativa*, 7:55–65.

Kalman, R. E. et al. (1960). Contributions to the theory of optimal control. *Bol. Soc. Mat. Mexicana*, 5(2):102–119.

Keerthi, S. and Gilbert, E. (1987). Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints. *Automatic Control, IEEE Transactions on*, 32(5):432–435.

Kerrigan, E. and Maciejowski, J. (2000a). Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control. *Proceedings of the 39th IEEE Conference on Decision and Control*, 5:4951–4956.

Kerrigan, E. and Maciejowski, J. (2000b). Soft constraints and exact penalty functions in Model Predictive Control. In *Proc. of the UKACC International Conference on Control*, Cambridge, UK.

Lao, L., Ellis, M., and Christofides, P. (2013). Proactive fault-tolerant model predictive control. *AIChE J.*, 59(8), 2810–2820.

Lee, J. H. and Yu, Z. (1997). Worst-case formulations of model predictive control for systems with bounded parameters. *Automatica*, 33:763–781.

Löfberg, J. (2004). Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan.

Maciejowski, J. M. (1999). Modelling and predictive control: Enabling technologies for reconfiguration. *Annual Reviews in Control*, 23:13–23.

Maciejowski, J. M. (2002). *Predictive Control: With Constraints.* Prentice Hall.

Magni, L. and Scattolini, R. (2007). Robustness and robust design of mpc for nonlinear discrete-time systems. In Findeisen, R., Allgöwer, F., and Biegler, L., editors, *Assessment and Future Directions of Nonlinear Model Predictive Control*, volume 358 of *Lecture Notes in Control and Information Sciences*, pages 239–254. Springer Berlin Heidelberg.

Marruedo, D. L., Alamo, T., and Camacho, E. (2002). Input-to-state stable mpc for constrained discrete-time nonlinear systems with bounded additive uncertainties. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 4, pages 4619–4624. IEEE.

Mayne, D., Rawlings, J., Rao, C., and Scokaert, P. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

Mayne, D. Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986.

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

Mellichamp, D. A., Edgar, T. F., Doyle, F. J., and Seborg, D. E. (2010). *Process Dynamics and Control*. John Wiley & Sons.

Miksch, T., Gambier, A., and Badreddin, E. (2008). Real-time implementation of fault-tolerant control using model predictive control. *World Congress*, pages 11136–11141.

Muller, M., Allgower, F., et al. (2012). Robustness of steady-state optimality in economic model predictive control. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference*, pages 1011–1016. IEEE.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer Science & Business Media.

Pietrzykowski, T. (1969). An exact potential method for constrained maxima. *SIAM Journal of Numerical Analysis*, 19(2):786–789.

Raimondo, D. M., Limon, D., Lazar, M., Magni, L., and ndez Camacho, E. F. (2009). Min-max Model Predictive Control of Nonlinear Systems: A Unifying Overview on Stability. *European Journal of Control*, 15(1):5–21.

Rakovic, S., Kerrigan, E., Kouramas, K., and Mayne, D. (2003). Approximation of the minimal robustly positively invariant set for discrete-time lti systems with persistent state disturbances. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 4, pages 1515–1520. IEEE.

Rao, C. V., Wright, S. J., and Rawlings, J. B. (1998). Application of Interior-Point Methods to Model Predictive Control. *Journal of Optimization Theory and Applications*, 99(3):723–757.

Richards, A. and How, J. (2006). Robust stable model predictive control with constraint tightening. In *American Control Conference, 2006*, number 4, pages 6–pp. IEEE.

Riverso, S., Battocchio, A., and Ferrari-Trecate, G. (2013). Pnpmpc toolbox.

Salfner, F. and Malek, M. (2007). Using hidden semi-Markov models for effective online failure prediction. In *Proc. of the IEEE Symp. on Reliable Distributed Systems*, pages 161–174.

Scibilia, F., Olaru, S., and Hovd, M. (2011). On feasible sets for MPC and their approximations. *Automatica*, 47(1):133–139.

Scokaert, P. O. M. and Rawlings, J. B. (1999). Feasibility issues in linear model predictive control. *AIChE Journal*, 45(8):1649–1659.

Siirola, J. J. and Edgar, T. F. (2012). Process energy systems: Control, economic, and sustainability objectives. *Computers and Chemical Engineering*, 47:134–144.

Yu, S., Bohm, C., Chen, H., and Allgower, F. (2010). Robust model predictive control with disturbance invariant sets. In *American Control Conference (ACC), 2010*, pages 6262–6267. IEEE.

Zhang, Y. and Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control*, 32(2):229–252.

Appendix A

MPC formulation

This chapter describes the compact MPC formulations used in this thesis.

A.1 Compact quadratic cost formulation

Consider the following MPC problem with quadratic cost:

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k && && \text{(A.1a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0,\dots,N-1\} && \text{(A.1b)}\\
& x_0 = x_{\text{init}}, && && \text{(A.1c)}\\
& x_k \in \mathbb{X}, && \forall k \in \{1,\dots,N-1\} && \text{(A.1d)}\\
& u_k \in \mathbb{U}, && \forall k \in \{0,\dots,N-1\} && \text{(A.1e)}\\
& x_N \in \mathcal{X}_{\text{terminal}}, && && \text{(A.1f)}
\end{aligned}
$$

where $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$ are the weights. The sets $\mathbb{X}$ and $\mathbb{U}$ are compact sets defining the constraints on the states and inputs. By recasting

$$x_{k+1} = Ax_k + Bu_k, \forall k \in \{0,\dots,N-1\} \tag{A.2}$$

to

$$x = \tilde{A}x_0 + \tilde{B}\mathbf{u}, \tag{A.3}$$

where

$$\tilde{A} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix},$$

$$\tilde{B} = \begin{bmatrix} B & & & & \\ AB & B & & & \\ A^2B & AB & B & & \\ \vdots & \vdots & \vdots & \ddots & \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \dots & B \end{bmatrix},$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix},$$

and

$$\mathbf{u} = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N-1} \end{bmatrix},$$

and using

$$\hat{Q} = \text{diag}(Q, Q, \dots, Q),$$

$$\hat{R} = \text{diag}(R, R, \dots, R),$$

the MPC problem can be reformulated to a dense QP problem of the form

$$\begin{aligned} \min_{u} \quad & J = 0.5\mathbf{u}^T H\mathbf{u} + x_0^T F\mathbf{u} \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0, \end{aligned} \tag{A.4}$$

where

$$H = \hat{B}^T\hat{Q}\hat{B} + \hat{R}, \tag{A.5}$$

$$F = \hat{A}^T\hat{Q}\hat{B} \tag{A.6}$$

And G, W, E are designed to include the constraints (A.1c)-(A.1f).

A.2 Compact linear cost formulation

Consider the MPC problem (A.1), but with a linear cost function

$$\sum_{k=0}^{N-1} l\left(x_k, u_k\right) = \sum_{k=0}^{N-1} \left(-qx_k + \|Ru_k\|_1\right), \tag{A.7}$$

Where $q \in \mathbb{R}^n$ and $R \in \mathbb{R}^{m \times m}$. This can be recast by using the matrices $\tilde{A}$ and $\tilde{B}$ from the previous section. The cost function is rewritten as

$$\sum_{k=0}^{N-1} \left(-qx_k + ru_k\right) = -\tilde{Q}\mathbf{x} + \tilde{R}\mathbf{u} = -\tilde{Q}(\tilde{A}x_0 + \tilde{B}\mathbf{u}) + \tilde{R}\mathbf{u} \tag{A.8}$$

$$= \left(\tilde{R} - \tilde{Q}\tilde{B}\right)\mathbf{u} - \tilde{Q}\tilde{A}x_0 \tag{A.9}$$

Where $r \in \mathbb{R}^m$ contains the diagonal elements of R,

$$\hat{Q} = \operatorname{diag}(q, q, \ldots, q),$$
$$\hat{R} = \operatorname{diag}(r, r, \ldots, r),$$

and $\mathbf{x}$, $\mathbf{u}$ is as defined in the previous section. Equation (A.8) can then be written as

$$\sum_{k=0}^{N-1} \left(-qx_k + ru_k\right) = H\mathbf{u} - Fx_0, \tag{A.10}$$

where

$$H = \tilde{R} - \tilde{Q}\tilde{B} \tag{A.11}$$

$$F = \tilde{Q}\tilde{A}. \tag{A.12}$$

Rewriting the ℓ_1 norm

In order to express the ℓ_1 norm in the cost function (A.7) in compact form, it needs to be rewritten using additional variables and constraints. Defining

$$u = u^+ - u^- \tag{A.13}$$

yields the following cost function with constraints

$$\sum_{k=0}^{N-1} l\left(x_k, u_k\right) = \sum_{k=0}^{N-1} \left(-qx_k + r(u_k^+ + u_k^-)\right), \tag{A.14}$$

$$u^+ \geq 0 \tag{A.15}$$

$$u^- \geq 0 \tag{A.16}$$

This yields the following compact form of the optimization problem (A.1) with the linear cost function (A.7).

$$\begin{aligned} \min_{\mathbf{u}} \quad & H\left(\mathbf{u}^+ + \mathbf{u}^-\right) - Fx_0 \\ \text{s.t.} \quad & x_0 = x_{\text{init}}, \\ & G\mathbf{u} \leq W + Ex_0, \\ & \mathbf{u} = \mathbf{u}^+ - \mathbf{u}^-, \\ & \mathbf{u}^+ \geq 0, \\ & \mathbf{u}^- \geq 0. \end{aligned} \tag{A.17}$$

Appendix B

KKT-conditions

Given the optimization problem

$$\begin{aligned} \min_{x} \quad & V(x) \\ \text{s.t.} \quad & G_{\mathrm{i}}(x) = 0 \quad \forall i \in \mathcal{E} \\ & G_{\mathrm{i}}(x) \leq 0 \quad \forall i \in \mathcal{I}. \end{aligned} \tag{B.1}$$

The solution x^* has to satisfy the Karush-Kuhn-Tucker (KKT) conditions.

Theorem B.1 (KKT-conditions, (Nocedal and Wright, 2006))**.** *Suppose that x^* is a local solution of* (B.1)*, that the function V and G_i in* (B.1) *are continuously differentiable and that the LICQ holds at x^*. Then there is a Lagrange multiplier vector λ^*, with components λ_i^*, $i \in \mathcal{E} \cup \mathcal{I}$ such that the following conditions are satisfied at (x^*, λ^*).*

$$\nabla_x \mathscr{L}(x^*, \lambda^*) = 0 \tag{B.2a}$$
$$G_e(x^*) = 0 \qquad \forall i \in \mathcal{E}, \tag{B.2b}$$
$$G_i(x^*) \leq 0 \qquad \forall i \in \mathcal{I}, \tag{B.2c}$$
$$\lambda_i^* \geq 0 \qquad \forall i \in \mathcal{I}, \tag{B.2d}$$
$$\lambda_i^* G_i(x^*) = 0 \qquad \forall i \in \mathcal{E} \cup \mathcal{I}. \tag{B.2e}$$

The KKT-conditions are first-order necessary conditions for a constrained optimization problem. However, if the problem is convex and regular which is the case in this thesis, they are also sufficient for a global optimum (Nocedal and Wright, 2006).

Appendix C

Set operations

Given two sets Ω and Φ, the Pontryagin difference is defined as

$$\Omega \sim \Phi \triangleq \{\omega \in \mathbb{R}^n \mid \omega + \phi \in \Omega,\ \forall \phi \in \Phi\} . \tag{C.1}$$

Similarly, the Minkowski sum is defined as

$$\Omega \oplus \Phi \triangleq \{\omega + \phi \mid \omega \in \Omega,\ \phi \in \Phi\} . \tag{C.2}$$

The multiplication of a set Ω by a matrix A denotes a mapping of all its elements

$$A\Omega \triangleq \{c \mid \exists \omega \in \Omega,\ c = A\omega\} . \tag{C.3}$$

Appendix D

Discrete minimal-time control

The minimal time required to steer a linear system from an initial given feasible state x_0, inside a compact set $\mathbb{S} = \{x|Hx \leq h\}$, with the initial state $x_0 \notin \mathbb{S}$, can be computed by the following MILP:

$$
\begin{aligned}
&\min && \sum_{k=1}^{N} -w_k y_k && && \text{(D.1a)}\\
&\text{s.t.} && x_{k+1} = Ax_k + Bu_k, && \forall k \in \{0, \ldots, N-1\} && \text{(D.1b)}\\
& && x_0 = \text{given}, && && \text{(D.1c)}\\
& && x_k \in \mathbb{X}, && \forall k \in \{0, \ldots, N-1\} && \text{(D.1d)}\\
& && u_k \in \mathbb{U}, && \forall k \in \{0, \ldots, N-1\} && \text{(D.1e)}\\
& && Hx_k \leq h + M(1 - y_k), && \forall k \in \{0, \ldots, N\} && \text{(D.1f)}\\
& && \sum_{k=0}^{N} y_k \geq 1, && && \text{(D.1g)}\\
& && y_k = \{0, 1\}. && \forall k \in \{0, \ldots, N\} && \text{(D.1h)}
\end{aligned}
$$

In (D.1), w_k is a sequence of positive, strictly increasing weights, e.g. $w_k := k$, and M is a big-M parameter. If (D.1) has an integer feasible solution, the minimum time $t^{\min}$ to get the state x_k inside the set $\mathbb{S}$ is given by the integer $k^{\min}$ for which the binary y_k first takes the value 1, i.e. $t^{\min} = \{k^{\min}|y_k = 1,\ \forall k \geq k^{\min}\}$. Observe that the negativity in (D.1a) ensures that the system stays in $\mathbb{S}$ for all positive times when first inside.

Appendix E

Conference paper

The following paper was submitted to the IEEE *Conference on Decision and Control* 2015.

An exact penalty-function approach to proactive fault-tolerant economic MPC

Brage Rugstad Knudsen and Jon H. Brusevold

Abstract— **Integration of fault tolerant control and receding horizon optimization is a powerful approach to include fault handling and dynamic economic optimization of a system. Proactive fault-tolerant model predictive control (FTMPC) seeks to utilize an estimated, conservative time window between the warning of an incipient fault in the system and the time at which the faulty component is rendered useless, to steer the state inside a recoverable region before the fault occurs. As such, a proactive FTMPC circumvents the issues of possible infeasibilities and destabilization often encountered in reactive approaches, while allowing the system to continue economic operation during the subsequent system repair. In this paper, we propose a proactive FTMPC scheme for incipient actuator faults by using an exact penalty function to steer the system inside an invariant set ensuring stability during the loss of actuation in the system. We consider the approach for linear systems operated with an economic MPC controller, thereby allowing the computation a lower bound for the penalty parameter to ensure exactness of the penalty function. We show nominal asymptotic stability of the proposed FTMPC scheme, and illustrate the approach by a numerical example.**

I. INTRODUCTION

Fault tolerance in dynamic system optimization is important both for the safety and economic optimality of the operations. A structured and robust fault-tolerant detection and control scheme enhances reliability and continuity of system operations, both for safety-critical processes and for chemical production and manufacturing [1]. Model predictive control (MPC) is an extensively used optimization-based control scheme, due to its ability to efficiently handle complex systems with hard control constraints and many inputs and outputs [2]. These features of MPC also allow for direct adaptation of the controller to faults in the system, by being an optimal-control scheme solved online [3]. Recently, however, there has been increased attention to economic model predictive control (EMPC), which contrary to separated real-time optimization (RTO) and MPC, merges dynamic economic operations with the feedback properties of conventional MPC [2], [4], [5], [6]. Combing fault-tolerance in an EMPC scheme hence is an attractive approach to design fault-tolerant, receding-horizon based economic operations of a system.

Active fault-tolerant control methods can broadly be classified as reactive or proactive methods [7]. Reactive approaches try to minimize the impact of a fault after it occurs, relying on a fault detection and isolation unit (FDI) and reconfiguration of the control system. *Proactive* fault-tolerant control methods employ an FDI unit to detect slowly developing, degradation of performance in process components, actuators or sensors that indicates an incipient fault [8], together with a probabilistic prediction method for the time of the incipient fault, e.g., [9], and, contrary to reactive methods, a proactive control action to prevent negative impact of the predicted fault situation. Proactive fault-tolerant methods is emerging as a complement to reactive schemes for designing robust and effective fault control, though it is important to emphasize that proactive approaches is intended only to supplement reactive schemes capable of handling abrupt faults. A proactive method can, on the other hand, be efficiently applied for maintaining process operation, minimize down time or prevent shut-downs in terms of certain types of faults, and also to perform scheduled maintenance.

In this paper, we focus on fault-tolerant MPC for handling incipient actuator faults. An actuator fault may cause loss of controllability at the current operating point and thereby destabilize the system. An alternative would be to operate the system within a set that guarantees feasibility for kinds of actuator faults. This latter approach, however, would generally be conservative and detrimental for the nominal economic operation of a system. A proactive FTMPC scheme for actuator faults [7], is obtained by allowing the system to operate outside the guaranteed stability region with one of the actuators inactive, but force the system inside this region upon indication of an incipient fault. The proactive FTMPC in [7] uses Lypanunov-based MPC with predesigned controllers to drive the system inside this safety system, while [10] develops a hybrid scheme with scenario based safety constraints and reconfigurable control. Further approaches that address actuator faults within an MPC scheme include among others [11], which is based on set-invariance concept to manage actuator fault occurrences, and the approach in [12] based on a bank of state estimators to match different fault situations. See also [1] and reference therein.

The proactive FTMPC scheme proposed in this paper builds upon [7], but applies EMPC with an exact penalty function to steer the state inside a safety set upon warning of an incipient actuator fault. A rule for switching between MPC problems for nominal, safe-mode transition and faulty operations is designed, based on the assumption of a separate available FDI unit to indicate and distinguish incipient and actuals fault. The proposed scheme allows for offline computation of a lower bound for the penalty parameter, thereby preventing unnecessary aggressive and

Both authors are with Department of Engineering Cybernetics, Norwegian University of Science and Technology. Corresponding author: Brage R. Knudsen (brage.knudsen@itk.ntnu.no) Support from NFR grant 228460/030 and from Cybernetica AS is gratefully acknowledged.

violent control action to steer the system inside the safety region, and allowing the system to retain economic operation during the time of loss in actuation. The paper is organized as follows: In Section II we present the problem and the set-up of the proposed proactive FTMPC scheme. Section II-A describes the computation of penalty parameters, and Section III analysis the stability properties of the controller. Section IV presents a numerical example to illustrate the proposed scheme, while Section V ends the paper with concluding remarks.

II. Problem Description

We consider proactive fault-tolerant MPC for linear discrete-time systems,

$$x_{k+1} = Ax_k + Bu_k, \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the system state, $u_k \in \mathbb{R}^m$ with $m > 1$ is the input, and where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. During nominal operations, the economics of the system are optimized by solving at each sample time t the economic optimization problem $P^{\text{nom}}(x)$:

$$V_N^{\text{nom}}(x) = \min \sum_{k=0}^{N-1} l(x_k, u_k) \tag{2a}$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k, \qquad k = 0, \dots, N-1 \tag{2b}$$

$$x_0 = x, \tag{2c}$$

$$(x_k, u_k) \in \mathbb{Y}^{\text{nom}}, \qquad k = 0, \dots, N-1 \tag{2d}$$

$$x_N = x_{\text{s}}^{\text{nom}}, \tag{2e}$$

where x is the current state of the system, and where the set

$$\mathbb{Y}^{\text{nom}} = \{(x,u) \in \mathbb{R}^{n\times m} | Dx + Hu \leq d\} \tag{3}$$

defines point-wise in time polytopic constraints on the states and inputs, including lower and upper bounds on the states and inputs (i.e. $\mathbb{Y}^{\text{nom}}$ is a compact set). Furthermore, $l(x,u)$ is an economic stage cost function, which we assume to be convex. The terminal equality constraint (2e) is defined by the solution $(x_{\text{s}}^{\text{nom}}, u_{\text{s}}^{\text{nom}})$ of the corresponding steady-state problem

$$\min\{l(x,u) \mid x = Ax + Bu, \ (x,u) \in \mathbb{Y}^{\text{nom}}\}. \tag{4}$$

We assume that $(x_{\text{s}}^{\text{nom}}, u_{\text{s}}^{\text{nom}})$ uniquely solves (4) with objective value $l(x_{\text{s}}^{\text{nom}}, u_{\text{s}}^{\text{nom}})$. Operation of the system in faulty and nominal mode imposes different state and/or input constraints, as well as modified control matrix B_j if actuator j is rendered useless. These varying constraints gives different optimal steady-state points. Let $\mathbf{u} = (u_0, u_1, \dots, u_{N-1})$ denote a feasible input sequence for (2). The set $\mathbb{X}_N^{\text{nom}}$ of admissible states for $P^{\text{nom}}(x)$ is then obtained by projecting the set of admissible inputs and initial states $\mathbb{Z}_N^{\text{nom}} = \{(x_0, \mathbf{u}) \mid \exists\, x_1, \dots, x_N \text{ satisfying (2b)–(2e)}\}$ onto $\mathbb{R}^n$. We allow the system (1) to be unstable, but we make the following stabilizability assumption.

Assumption 1. The nominal system (A,B) and the faulty system (A,B_j) are controllable, and N is chosen sufficiently large such that all *admissible* initial states x can be steered to an admissible economic steady-state point $(x_{\text{s}}', u_{\text{s}}')$ within N steps will satisfying the given state and input constraints.

Assumption 1 ensures that the system can be steered from any admissible initial state x to an admissible steady-state x_{s}' in N timesteps. It is important to emphasize that we assume this condition to hold for *any* admissible economic steady-state point, as the latter is changed by the introduction of safety constraints. Corresponding to the conventional MPC control law, only the first move $u_{\text{e}}^{\text{nom}}(x) := u_0^*$ of the optimal input sequence of $\mathbf{u}^*$ is applied to the system. The finite-horizon problem (2) is then repeatedly reoptimized in a receding horizon manner with current state (2c) updated through measurements of x. Note that the set $\mathbb{X}_N^{\text{nom}}$ is positively (or forward) invariant due to the imposed terminal equality constraint (2e), i.e. $x \in \mathbb{X}_N^{\text{nom}}$ implies $(Ax + Bu_{\text{e}}^{\text{nom}}(x)) \in \mathbb{X}_N^{\text{nom}}$ [6].

Our objective is to construct a proactive fault-tolerant economic MPC controller that allows continued (suboptimal) economic operations of the system in the presence of an incipient actuator fault. The following proactive scheme is considered: At time t', an FDI unit indicates an incipient fault in actuator $j \in \{1 \dots m\}$ and a conservative estimate t_{f} of the time when the fault will occur. For simplicity, we consider only the fault scenario where the controller is rendered completely inactive, although the scheme can easily be extended to fault scenarios where the actuator loses a fraction of its maximum actuation. In the time window $t_{\text{f}} - t'$, the MPC controller must steer the system from its nominal economic steady-state point $x_{\text{s}}^{\text{nom}}$ to a controllable set $\mathbb{S}_j \subset \mathbb{R}^n$ with actuator j inactive as illustrated in Fig. 1. In [7], this is performed by assuming that (Lyapunov-based) controllers, $u = h_0(x)$ and $u = h_j(x)$, can be designed such that $u = h_0(x)$ first drives the state inside the stabilizable region within t_{f}, while $u = h_j(x)$ subsequently stabilizes asymptotically the origin of the faulty closed-loop system.

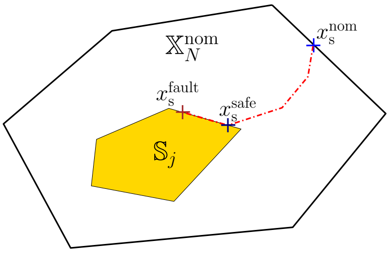

$\mathbb{X}_N^{\text{nom}}$
$x_{\text{s}}^{\text{nom}}$
$x_{\text{s}}^{\text{fault}}$
$x_{\text{s}}^{\text{safe}}$
$\mathbb{S}_j$

Fig. 1. Schematic illustration of proposed proactive FTMPC scheme.

To steer the system state x_k inside a *safe*, controllable set within the estimated time t_{f} of failure of actuator j we propose to use exact penalty functions [13]. Let

$$\mathbb{S}_j = \{x \mid G_j x \leq f_j\} \tag{5}$$

define a controllable safety set $\mathbb{S}_j$ with actuator j rendered inactive, with $G_j \in \mathbb{R}^{q\times n}$. This set can either be defined by operators of the plant or the system, in terms of known, conservative safety constraints on the state or a set of controlled variables, or it may defined by the N-step stabilizable set for $x_{\mathrm{s}}^{\mathrm{fault}}$ subject to the constraints by an actuator fault, which is control invariant and a subset of the maximum control invariant set with the same constraints, see [14], [15]. We will focus on latter definition of $\mathbb{S}_j$, and assume this can be computed by for instance the method in [16]. The set $\mathbb{S}_j$ will normally be a strict subset of $\mathbb{X}_N^{\mathrm{nom}}$ and thereby render (2) infeasible when operating at steady state $x_{\mathrm{s}}^{\mathrm{nom}}$ if imposed directly as constraints in $P^{\mathrm{nom}}(x)$ at time t' for all k. Hence, we must impose the constraints (5) through a penalty function, or equivalently, through soft constraints with a penalty norm.

At time t' when the MPC controller receives information about an incipient fault in actuator j, we switch to a transition problem $P^{\mathrm{safe}}(x)$ to drive the system to a safe mode:

$$
\begin{aligned}
V_N^{\mathrm{safe}}(x) = \min \sum_{k=0}^{N-1} l(x_k,u_k) + \mu\,||\varepsilon||_1 & & \text{(6a)}\\
\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k, \quad & k=0,\dots,N-1 & \text{(6b)}\\
x_0 = x, \quad & & \text{(6c)}\\
(x_k,u_k) \in \mathbb{Y}^{\mathrm{nom}}, \quad & k=0,\dots,N-1 & \text{(6d)}\\
G_j x_k \le f_j + \varepsilon_k, \quad & k=0,\dots,N-1 & \text{(6e)}\\
\varepsilon_k \ge 0, \quad & k=0,\dots,N-1 & \text{(6f)}\\
x_N = x_{\mathrm{s}}^{\mathrm{safe}} \quad & & \text{(6g)}
\end{aligned}
$$

where ε_k are time-varying, nonnegative q-dimensional slack variables, penalized by the ℓ_1 penalty function with penalty parameter $\mu > 0$. We confine the penalty function to the ℓ_1 norm, while the scheme may be extended to any ℓ_p norm subject to certain modifications. The new steady-state point $x_{\mathrm{s}}^{\mathrm{safe}}$ must satisfy $x_{\mathrm{s}}^{\mathrm{safe}} \in \mathbb{S}_j$, and is obtained by solving the constrained steady-state problem

$$
\min\{l(x,u) \mid x = Ax + Bu,\ (x,u) \in \mathbb{Y}^{\mathrm{nom}} \cap \mathbb{S}_j\}. \qquad (7)
$$

We denote the optimal objective value of (7) $l(x_{\mathrm{s}}^{\mathrm{safe}}, u_{\mathrm{s}}^{\mathrm{safe}})$. Solving $P^{\mathrm{safe}}(x)$ in a receding horizon defines the implicit feedback control law $u_{\mathrm{e}}^{\mathrm{safe}}(x) := u_0^{\mathrm{safe}}$ as described for $P^{\mathrm{nom}}(x)$ above.

The soft-constraint formulation (6) is equivalent with optimizing the nonsmooth penalty function $\min \sum_{k=0}^{N-1} l(x_k,u_k) + \mu \sum ||\max(0, c_{\mathscr{I}}(x))||_1$ subject to the remaining constraints in (6), where $c_{\mathscr{I}}$ is a vector-function representation of the constraints (5) written in non-positive inequality form. A penalty function $F(x,\mu)$ is termed *exact* if, for a parameter choice $\mu > \mu^*$ where $\mu^* > 0$ is a certain threshold value, the solution of an unconstrained problem $\min_x F(x,\mu)$ is either a KKT point of the original constrained problem, or infeasible stationary points [13], [17]. The same condition for exactness of a penalty function continuous to hold for a smooth, constrained reformulation as in (6), see [18], [19]. For reformulated penalty functions with slack variables and soft constraints, exactness of the penalty function hence implies that the soft and the hard constrained problem only differs if the hard-constrained problem is infeasible [20].

Proposition 1. If Assumption 1 holds, and $\mu > \mu^*$, where μ^* is a lower threshold value to ensure that the penalty function is exact, then the solution $(\mathbf{x}^*, \mathbf{u}^*, \varepsilon^*)$ to the reformulated ℓ_1 exact penalty function $P^{\mathrm{safe}}(x)$ will steer the state x_k inside $\mathbb{S}_j$ in the minimum number of timesteps.

Proof. With a sufficiently large penalty parameter $\mu > \mu^*$, a feasible solution $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ to the soft constrained problem (6), obtained by reformulation of an exact, non-differentiable penalty function, satisfies the KKT conditions of the corresponding hard-constrained problem if $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ is a feasible solution to this problem. If $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, 0)$ is infeasible for (6), the exactness of the penalty function (6a) will ensure constraint satisfaction, i.e. the KKT condition, for those constraints in (6e) that can be satisfied, i.e. the time-varying slack variables $\varepsilon_k^* = 0, \forall k \ge \bar{k}$ for some $\bar{k} > 0$. Consequently, the ℓ_1 exact penalty function will yield $\varepsilon_k^* > 0$ only for those k in (6e) that would yield infeasibility for the hard constraint $G_j x_k \le f_j$, thereby ensuring that these constraints are violated only if necessary, and hence in the minimum number of timesteps $\bar{k}$. The assumption of convexity of the $l(x,u)$ ensures global optimality of the solution $(\mathbf{x}^*, \mathbf{u}^*, \varepsilon^*)$. □

Conditions for exactness of penalty function and a technique for computing μ is given in Section II-A. For the proposed proactive FTMPC approach, we distinguish between two scenarios relating the estimated fault time t_{f} to the prediction horizon N: If $t_{\mathrm{f}} > t' + N$, then feasibility of $P^{\mathrm{safe}}(x)$ at time t' will ensure $x_k \in \mathbb{S}_j$ within t_{f}. Else, if $t_{\mathrm{f}} \le t' + N$ we must include a check of ε^* from the solution of $P^{\mathrm{safe}}(x)$ at time t'. Let $\varepsilon_{t_{\mathrm{f}}|t'}^*$ be the value of slack vector ε_k^* at prediction time $k = t_{\mathrm{f}} - t'$ computed at sample time t'. If $\varepsilon_{t_{\mathrm{f}}|t'}^* > 0$, the state cannot reach $\mathbb{S}_j$ within the estimated time t_{f} of actuator fault, in which the system must be shut down or switched to an emergency mode. Otherwise, $\varepsilon_{t_{\mathrm{f}}|t'}^* = 0$, and the state is steered inside $\mathbb{S}_j$ within time t_{f}.

Remark 1. Enforcing hard constraints $G_j x_k \le f_j$ for $k \ge t_{\mathrm{f}} - t'$ would not change the optimal solution $(\mathbf{x}^*, \mathbf{u}^*, \varepsilon^*)$ when the penalty function is exact. If $\varepsilon_k^* = 0, \forall k \ge t_{\mathrm{f}} - t'$ is a feasible solution to $\mathrm{P}^{\mathrm{safe}}(x)$, then exactness of the penalty function will ensure this indeed is the solution to $\mathrm{P}^{\mathrm{safe}}(x)$.

Unless the system must shut-down due to failure of the MPC controller with $P^{\mathrm{safe}}(x)$ to steer the system state inside $\mathbb{S}_j$ before the estimated failure time of actuator j, the MPC controller can continue economic operation of the system inside the safety set until the fault occurs. We assume that the FDI unit alerts the MPC controller when (if) the actuator *actually* fails or is taken out of operation to be replaced. At this time instant, denoted t_{fa}, the LTI model and input constraints are updated to account for the loss of actuation, in which the MPC controller switches to the optimization problem $P^{\mathrm{fault}}(x)$:

$$
\begin{aligned}
V_N^{\mathrm{fault}}(x) = \min \sum_{k=0}^{N-1} l(x_k,u_k) & & \text{(8a)}\\
\text{s.t.} \quad x_{k+1} = Ax_k + B_j u_k, \quad & k=0,\dots,N-1 & \text{(8b)}
\end{aligned}
$$

$$x_0 = x, \tag{8c}$$

$$(x_k, u_k) \in \mathbb{Y}_j, \qquad k = 0, \dots, N-1 \tag{8d}$$

$$G_j x_k \leq f_j, \qquad k = 0, \dots, N-1 \tag{8e}$$

$$x_N = x_{\mathrm{s}}^{\mathrm{fault}} \tag{8f}$$

$\mathbb{Y}_j$ contains updated input constraints, and the new optimal steady-state point, $x_{\mathrm{s}}^{\mathrm{fault}}$, is computed correspondingly to (4) and (7),

$$\min\{l(x,u) \mid x = Ax + B_j u,\ (x,u) \in \mathbb{Y}_j \cap \mathbb{S}_j\}. \tag{9}$$

with optimal objective value $l(x_{\mathrm{s}}^{\mathrm{fault}}, u_{\mathrm{s}}^{\mathrm{fault}})$. Observe that $x_{\mathrm{s}}^{\mathrm{fault}} \neq x_{\mathrm{s}}^{\mathrm{safe}}$ only if $u_{\mathrm{s}}^{\mathrm{safe}} \neq 0$ computed by (7). We define the implicit feedback control law obtained by solving $P^{\mathrm{fault}}(x)$ in receding horizon fashion for $u_{\mathrm{e}}^{\mathrm{fault}}(x) := u_0^{\mathrm{fault}}$ as described for $P^{\mathrm{nom}}(x)$ above. The EMPC controller operates the system in this fault-updated safe mode until the faulty actor has been replaced or inspected, in which nominal economic operations of the system is retained by switching to solving $P^{\mathrm{nom}}(x)$ in each sample time t.

A. Computing the penalty parameter

A critical criteria for the proposed approach is that the system enters the safety set $\mathbb{S}_j$ before the fault occurs. This requires the penalty function (6) to be exact, which means that the solution to the soft and the hard constrained problem only differs if the hard-constrained problem is infeasible. Selecting a numerical value for μ may be difficult. It is generally undesirable to assign an arbitrary high value to this parameter to ensure exactness of the penalty function, as this may lead to violent control action and possibly be harmful to the actuators [21]. We therefore seek to find a lower bound on μ in order to guarantee that the penalty function is exact. We first introduce the concept of the dual norm which is an essential component in the theory of exact penalty functions.

For every norm $\|\cdot\|$, there exists a dual norm, $\|\cdot\|_{\mathrm{D}}$, defined as [22]

$$||v||_{\mathrm{D}} = \max_{||\lambda|| \leq 1} v'\lambda \tag{10}$$

In particular, the dual of $\|\cdot\|_1$ is $\|\cdot\|_\infty$, and the dual of $\|\cdot\|_\infty$ is $\|\cdot\|_1$. A well-known result for the exactness of penalty functions is that the penalty parameter, μ, needs to be larger than the value of the dual norm of the Lagrangian multiplier for the hard constrained problem [22, Th. 14.3.1]. For exactness of the ℓ_1 penalty function, the penalty function is hence exact if $\mu > \mu^* = \max \|\lambda\|_\infty$. Consequently, to compute μ^* for (6a), the maximum value of the ℓ_∞ norm of the Lagrangian multipliers for the hard-constrained problem for all initial states $x \in \mathbb{S}_j$ needs to be calculated. To this end, we use the approach in [21], which is based on reformulation of a bi-level program to a mixed-integer linear programming (MILP) for computing μ^*. Note that these computation must be performed for each safety set $\mathbb{S}_j$ for faults in different actuators j, however, by an offline computation. Furthermore, note that we only need to include the Lagrangian multipliers for the constraints that are softened [19].

III. Stability

Nominal stability of EMPC has been proved for systems with a terminal equality constraint, satisfying strong duality [6] or strict dissipativity [5], or with a terminal cost and inequality constraints for systems satisfying strict dissipativity [4]. In this paper, we base the stability proof on the approach in [6], and show only for the ℓ_1 penalty function.

Assumption 2. If $l(x,u)$ contains other than linear terms, these must be strictly convex, and a constraint qualification, e.g. Slater's condition, must additionally be satisfied at the optimal steady-state point.

If $l(x,u)$ is a linear, economic objective function, the MPC problems (2), (6) and (8) resort to linear programs (LPs), in which strong duality holds [23, Ch. 5]. If $l(x,u)$ is quadratic, e.g. $u_k^T R u_k$, then R must be positive definite. The additional assumption of a constraint qualification assures strong duality to hold at optimal steady state.

To analyze the stability properties of the proposed FTMPC scheme we introduce "rotated" stage costs [6],

$$L^{\mathrm{nom}}(x,u) = l(x,u) + (x - Ax - Bu)'\lambda_{\mathrm{s}}^{\mathrm{nom}} - l(x_{\mathrm{s}}^{\mathrm{nom}}, u_{\mathrm{s}}^{\mathrm{nom}}), \tag{11a}$$

$$L^{\mathrm{safe}}(x,u,\varepsilon) = l^{\mathrm{safe}}(x,u,\varepsilon) + (x - Ax - Bu)'\lambda_{\mathrm{s}}^{\mathrm{safe}} - l(x_{\mathrm{s}}^{\mathrm{safe}}, u_{\mathrm{s}}^{\mathrm{safe}}), \tag{11b}$$

$$L^{\mathrm{fault}}(x,u) = l(x,u) + (x - Ax - B_j u)'\lambda_{\mathrm{s}}^{\mathrm{fault}} - l(x_{\mathrm{s}}^{\mathrm{fault}}, u_{\mathrm{s}}^{\mathrm{fault}}), \tag{11c}$$

where $l^{\mathrm{safe}}(x,u,\varepsilon) := l(x_k,u_k) + \mu \sum_{i=1}^{q} \varepsilon_{ik}$ is the point-wise in time stage cost (6a) as a function of x, u and ε with ℓ_1 penalty. Moreover, $\lambda_{\mathrm{s}}^{\mathrm{nom}}, \lambda_{\mathrm{s}}^{\mathrm{safe}}$ and $\lambda_{\mathrm{s}}^{\mathrm{fault}}$ are Lagrangian multipliers for the LTI steady-state model such that strong duality holds for the three steady-state problems (4), (7) and (9), respectively. Note that strong duality holds by Assumption 2, and that by allowing slack on the constraint $G_j x \leq f_j$ only up to $N-1$, the steady-state problem of $P^{\mathrm{safe}}(x)$ is independent of ε.

Lemma 1. The following relates the rotated costs (11) and the respective MPC problems:

1) Solving $\mathrm{P}^{\mathrm{nom}}(x)$ in (2) with objective (2a) replaced with $\tilde{V}_N^{\mathrm{nom}}(x) = \min \sum_{k=0}^{N-1} L^{\mathrm{nom}}(x_k,u_k)$ gives equal solution.
2) Solving $\mathrm{P}^{\mathrm{safe}}(x)$ in (6) with the objective (6a) replaced with $\tilde{V}_N^{\mathrm{safe}}(x) = \min \sum_{k=0}^{N-1} L^{\mathrm{safe}}(x_k,u_k,\varepsilon_k)$ gives equal solution.
3) Solving $\mathrm{P}^{\mathrm{fault}}(x)$ in (8) with the objective (8a) replaced with $\tilde{V}_N^{\mathrm{fault}}(x) = \min \sum_{k=0}^{N-1} L^{\mathrm{fault}}(x_k,u_k)$ gives equal solution.

Proof. All the three rotated costs are point-wise in time summed from $k = 0$ to $N-1$, and the respective MPC optimization problems have terminal equality constraint. The results hence follows immediately from Lemma 2 in [6]. □

The above lemma is used directly to prove *nominal* stability of the proposed proactive FTMPC scheme, that is, for nominal model and no disturbances.

Theorem 1. *(Nominal stability)*: If Assumption 1 and 2 hold, and $\mu > \mu^*$ such that the ℓ_1-penalty function in (6) is exact, then the following stability properties hold:

1) *(Nominal operations)*: $x_{\rm s}^{\rm nom}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + Bu_{\rm e}^{\rm nom}(x)$ with Lyapunov function $\tilde{V}_N^{\rm nom}(x)$ and region of attraction $\mathbb{X}_N^{\rm nom}$.
2) *(Safe operations)*: At time t', if (a) $t' + N \le t_{\rm f}$ and $\varepsilon_{t_{\rm f}|t'} = 0$, or (b) if $t_{\rm f} > t' + N$, the system will be steered inside the safety set within the $t_{\rm f}$, in which $x_{\rm s}^{\rm safe}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + Bu_{\rm e}^{\rm safe}(x)$ with Lyapunov function $\tilde{V}_N^{\rm safe}(x)$ and region of attraction $\mathbb{X}_N^{\rm nom}$.
3) *(Fault operations)*: $x_{\rm s}^{\rm fault}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + Bu_{\rm e}^{\rm fault}(x)$ with Lyapunov function $\tilde{V}_N^{\rm fault}(x)$ and region of attraction $\mathbb{S}_j$

Proof. A sketch of the proof is given for the three parts individually. Part 1): Recursive feasibility of $P^{\rm nom}(x)$ is ensured by the terminal equality constraint $x_N = x_{\rm s}^{\rm nom}$ and Assumption 1. Furthermore, Assumption 2 ensures strong duality to hold at steady state $x_{\rm s}^{\rm nom}$. It can hence be verified that $\tilde{V}_N^{\rm nom}(x)$ satisfies the properties of a Lyapunov function [6, Th. 1], and in particular that

$$\tilde{V}_N^{\rm nom}(Ax + Bu_{\rm e}^{\rm nom}(x)) \le \tilde{V}_N^{\rm nom}(x) - L^{\rm nom}(x, u_{\rm e}^{\rm nom}(x)) \quad \text{(12a)}$$
$$\le \tilde{V}_N^{\rm nom}(x) - \beta(|x - x_{\rm s}^{\rm nom}|)) \quad \text{(12b)}$$

for all $x \in \mathbb{X}_N^{\rm nom}$, and for a K_∞-function $\beta(\cdot)$. This proves part 1) of the theorem.

Part 2): Let $0 < \bar{k} \le t_{\rm f} - t'$ be an integer, such that $\varepsilon^*_{k|t'} = 0$ for all $k \ge \bar{k}$. At sample time t', let $\{\varepsilon_{0|t'}, \varepsilon_{1|t'}, \dots, \varepsilon_{\bar{k}-1|t'}, 0, \dots, 0\}$ be a feasible sequence of slack variables, and let $\mathbf{u}$ a feasible control sequence. Applying the feedback control law $u_{\rm e}^{\rm safe}(x)$ at time t', then at time $t'+1$, the sequence $\{\varepsilon_{1|t'}, \dots, \varepsilon_{\bar{k}-1|t'}, 0, 0, \dots, 0\}$ and $\{u_1, u_2, \dots, u_{N-1}, u_{\rm s}^{\rm safe}\}$ will be feasible with $\tilde{x} = Ax + Bu_{\rm e}^{\rm safe}(x)$ as initial condition. This follows from the terminal equality constraint (6g) and by requiring zero slack on the constraints $G_j x \le f_j$ at the end of the horizon. Feasibility of $P^{\rm safe}(x)$ for all sample times $t \ge t'$ and for all initial states $x \in \mathbb{X}_N^{\rm nom}$ follows by induction.

For the two scenarios of $t_{\rm f}$ relative to N, the following holds; (a) If $t' + N \le t_{\rm f}$ and $\varepsilon^*_{t_{\rm f}|t'} = 0$, then by the recursive feasibility, exactness of the penalty term, and Proposition 1, the number of positive slack vectors will decrease by one for each receding horizon iteration, decreasing the total magnitude of the ℓ_1 penalty term. Hence if $\varepsilon^*_{t_{\rm f}|t'} = 0$, then x_k will be steered $\mathbb{S}_j$ within $t_{\rm f}$, and indeed $x \in \mathbb{S}_j$ for all sample times $t \ge t_{\rm f}$ due to the positive invariance of $\mathbb{S}_j$. If $t_{\rm f} > t' + N$, then it follows immediately that $x_k \in \mathbb{S}_j$ within time $t_{\rm f}$ by feasibility of $P^{\rm safe}(x)$ at sample time t', and by the same arguments as above. Asymptotic stability of $x_{\rm s}^{\rm safe}$ from switching to $P^{\rm safe}(x)$ at time t' can then be established by using $\tilde{V}_N^{\rm safe}(x)$ for all $x \in \mathbb{X}_N^{\rm nom}$, and establishing an inequality similar to (12) with $L^{\rm safe}(x_k, u_k, \varepsilon_k)$ and a K_∞-function $\tilde{\beta}(\cdot)$.

Part 3): If the MPC problem $P^{\rm safe}(x)$ with control law $u_{\rm e}^{\rm safe}(x)$ is able to steer the system state x_k inside $\mathbb{S}_j$ within time $t_{\rm f}$, then for all initial states $x \in \mathbb{S}_j$, using the same arguments as in part 1) and in [6, Th. 1], it holds that $x_{\rm s}^{\rm fault}$ is an asymptotically stable steady-state point of the closed-loop system $x_{k+1} = Ax_k + Bu_{\rm e}^{\rm fault}(x)$ with region of attraction $\mathbb{S}_j$. □

We comment that a asymptotic stability of $P^{\rm safe}(x)$ may also be achieved by using the approach in [4].

IV. NUMERICAL EXAMPLE

In this section, we illustrate the proposed FTMPC scheme by a two-dimensional example. The N-step stabilizable sets are calculated using toolbox available at [1]. All simulations are performed in YALMIP [24], while CPLEX is used to solve the MILP for defining μ. The LTI system is open-loop unstable and defined by the matrices

$$A = \begin{bmatrix} 1.3337 & 0.9443 \\ 0.5902 & 1.3337 \end{bmatrix}, \; B = \begin{bmatrix} -0.2572 & -0.3817 \\ -0.2665 & -0.1954 \end{bmatrix}, \quad (13)$$

and designed with the economic objective function

$$\sum_{k=0}^{N-1} l(x_k, u_k) = \sum_{k=0}^{N-1} (-qx_k + ru_k), \quad (14)$$

where $N = 10$, $q = \begin{bmatrix} 10 & 10 \end{bmatrix}$, $r = \begin{bmatrix} 3 & 1 \end{bmatrix}$. The constraints on x and u are

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \le \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} 6 \\ 6 \end{bmatrix}, \; \begin{bmatrix} 0 \\ 0 \end{bmatrix} \le \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \le \begin{bmatrix} 5 \\ 15 \end{bmatrix}. \quad (15)$$

We will consider the scenario where the actuator u_2 has a dropout, i.e. $u_2 = 0$ for $t_f \le t < t_{\rm fix}$. The MPC is warned about the incipient fault at $t' = 20$, at $t_{\rm f} = 40$ the fault hits the system and u_2 is rendered unusable. Finally, the fault is repaired at $t_{\rm fix} = 60$. From the computation of $\mu^* = \max_{x \in \mathbb{S}_j} \|\lambda\|_\infty$ by the approach outlined Section II-A, we set $\mu = 20$.

Fig. 2 shows the system trajectory by using the approach described in Section II-A. The system operates in the economic optimal point, $x_{\rm s}^{\rm nom} = \begin{bmatrix} 6 & 3.75 \end{bmatrix}^T$, until the controller receives information about the upcoming fault. The system is then driven into $\mathbb{S}_j$, and reaches the temporary steady-state point $x_{\rm s}^{\rm safe} = \begin{bmatrix} 0.6043 & 1.6960 \end{bmatrix}^T$. Observe that it is crucial to compute $\max_{x \in \mathbb{S}_j} \|\lambda\|_\infty$, since a smaller μ might not guarantee that the system is driven into the set $\mathbb{S}_j$, where the MPC controller retains feasible when the system is affected by the fault. When the fault occurs, the MPC is switched to $P^{\rm fault}(x)$, in which the faulty model is updated and the system is steered to the new economic optimal steady-state point, $x_{\rm s}^{\rm fault} = \begin{bmatrix} 1.8580 & 0.7048 \end{bmatrix}^T$. At sample time $t = t_{\rm fix}$, actuator u_2 is fixed and nominal operation is resumed, and the system is driven back to its original optimal steady-state point $x_{\rm s}^{\rm nom}$. We compare our approach with an open-loop discrete minimal-time control (DMTC) computed by (16) in the Appendix. The two approach uses equal number of timesteps to reach $\mathbb{S}_j$, while it can seen that the minimal-time

[1] http://www-control.eng.cam.ac.uk/eck21/matlab/invsetbox/

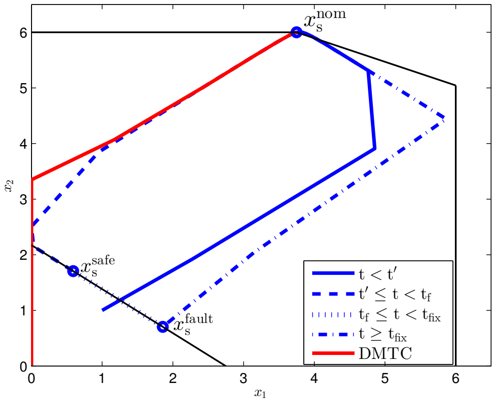


Fig. 2. State-trajectory showing the different phases of the system evolution, as well as the optimal steady-state points. The outer area represents the feasible set for nominal operation and the inner area (triangle) represents the feasible set after the fault, $\mathbb{S}_j$. Warning about incipient fault is given at sample time t', the fault occurs at time t_f and is repaired by time t_fix. The red line shows the state trajectory for an open-loop, discrete minimum time control (DMTC) to reach $\mathbb{S}_j$, computed by (16) in the Appendix.

approach renders a different trajectory. This follows from the well-known property that discrete minimal-time control is not necessarily bang-bang.

V. Concluding Remarks

In this paper, a novel proactive FTMPC scheme for incipient actuator faults based on exact penalty functions has been presented. The EMPC-based control scheme was shown to provide asymptotic stability of the economic steady-state points provided that steering the system inside the safety set within the estimated fault time actually is feasible. Further research includes extending the scheme to be robust in terms of handling disturbances on the system.

References

[1] Y. Zhang and J. Jiang, “Bibliographical review on reconfigurable fault-tolerant control systems,” *Annual Reviews in Control*, vol. 32, no. 2, pp. 229–252, 2008.

[2] D. Q. Mayne, “Model predictive control: Recent developments and future promise,” *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[3] J. M. Maciejowski, “Modelling and predictive control: Enabling technologies for reconfiguration,” *Annual Reviews in Control*, vol. 23, pp. 13–23, 1999.

[4] R. Amrit, J. B. Rawlings, and D. Angeli, “Economic optimization using model predictive control with a terminal cost,” *Annual Reviews in Control*, vol. 35, no. 2, pp. 178–186, 2011.

[5] D. Angeli, R. Amrit, and J. B. Rawlings, “On average performance and stability of economic model predictive control,” *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1615–1626, 2012.

[6] M. Diehl, R. Amrit, and J. B. Rawlings, “A Lyapunov function for economic optimizing model predictive control,” *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 703–707, 2011.

[7] L. Lao, M. Ellis, and P. D. Christofides, “Proactive fault-tolerant model predictive control,” *AIChE Journal*, vol. 59, no. 8, 2013.

[8] M. A. Demetriou and M. P. Polycarpou, “Incipient fault diagnosis of dynamical systems using online approximators,” *IEEE Transactions on Automatic Control*, vol. 43, no. 11, pp. 1612–1617, 1998.

[9] F. Salfner and M. Malek, “Using hidden semi-Markov models for effective online failure prediction,” in *Proc. of the IEEE Symp. on Reliable Distributed Systems*, 2007, pp. 161–174.

[10] T. I. Bø and T. A. Johansen, “Dynamic Safety Constraints by Scenario Based Economic Model Predictive Control,” in *IFAC World congress*, no. 2009, 2014.

[11] G. Franze, F. Tedesco, and D. Famularo, “Actuator Fault Tolerant Control: A Receding Horizon Set-Theoretic Approach,” *IEEE Transactions on Automatic Control*, 2014, (to appear).

[12] A. Yetendje, M. M. Seron, and J. A. D. Doná, “Robust multiactuator fault-tolerant MPC design for constrained systems,” *International Journal of Robust and Nonlinear Control*, vol. 23, no. 16, pp. 1828–1845, 2013.

[13] T. Pietrzykowski, “An exact potential method for constrained maxima,” *SIAM Journal of Numerical Analysis*, vol. 19, no. 2, pp. 786–789, 1969.

[14] E. Kerrigan and J. Maciejowski, “Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control,” in *Proc. of IEEE Conf. on Decision and Control*, 2000, pp. 4951–4956.

[15] F. Blanchini, “Set invariance in control,” *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[16] F. Scibilia, S. Olaru, and M. Hovd, “On feasible sets for MPC and their approximations,” *Automatica*, vol. 47, no. 1, pp. 133–139, 2011.

[17] G. Di Pillo and L. Grippo, “Exact penalty functions in constrained optimization,” *SIAM Journal on Control and Optimization*, vol. 27, no. 6, pp. 1333–1360, 1989.

[18] E. Kerrigan and J. Maciejowski, “Soft constraints and exact penalty functions in Model Predictive Control,” in *Proc. of the UKACC International Conference on Control*, Cambridge, UK, 2000.

[19] S. Janesch and L. Santos, “Exact penalty methods with constrained subproblems,” *Investigacióon Operativa*, vol. 7, pp. 55–65, 1997.

[20] P. O. M. Scokaert and J. B. Rawlings, “Feasibility issues in linear model predictive control,” *AIChE Journal*, vol. 45, no. 8, pp. 1649–1659, 1999.

[21] M. Hovd and F. Stoican, “On the design of exact penalty functions for MPC using mixed integer programming,” *Computers & Chemical Engineering*, vol. 70, no. 5, pp. 104–113, 2014.

[22] R. Fletcher, *Practical Methods of Optimization*, 2nd ed. John Wiley & Sons, 1987.

[23] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York: Cambridge University Press, 2004.

[24] J. Lø fberg, “YALMIP : A toolbox for modeling and optimization in MATLAB,” in *Proc.of the CACSD Conference*, Taipei, Taiwan, 2004.

Appendix

The minimal time required to steer a linear system from an initial given feasible state x_0, inside a compact set $\mathbb{S} = \{x|Gx \leq f\}$, with the initial state $x_0 \notin \mathbb{S}$, can be computed by the following MILP:

$$
\begin{aligned}
& \min \sum_{k=1}^{N} -w_k y_k && && \text{(16a)}\\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, && k = 0,\ldots,N-1 && \text{(16b)}\\
& x_0 = \text{given}, && && \text{(16c)}\\
& (x_k,u_k) \in \mathbb{Y}^{\mathrm{nom}}, && k = 0,\ldots,N-1 && \text{(16d)}\\
& Gx_k \leq f + M(1-y_k), && k = 1,\ldots,N && \text{(16e)}\\
& \sum_{k=0}^{N} y_k \geq 1 && && \text{(16f)}\\
& y_k = \{0,1\}, && k = 1,\ldots,N && \text{(16g)}
\end{aligned}
$$

In (16), w_k is a sequence of positive, strictly increasing weights, e.g. $w_k := k$, and M is a big-M parameter. If (16) has an integer feasible solution, the minimum time $t^{\min}$ to get the state x_k inside the set $\mathbb{S}$ is given by the integer $k^{\min}$ for which the binary y_k first takes the value 1, i.e. $t^{\min} = \{k^{\min}|y_k = 1,\ \forall k \geq k^{\min}\}$. Observe that the negativity in (16a) ensures that the system stays in $\mathbb{S}$ for all positive times when first inside.