ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

ADAPTIVE CONTROL VECTOR PARAMETERIZATION AND MOVE BLOCKING STRATEGIES IN MODEL-PREDICTIVE CONTROL

M.Sc. THESIS

Canan DOMBAYCI

Department of Chemical Engineering

Chemical Engineering Programme

DECEMBER 2013

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

ADAPTIVE CONTROL VECTOR PARAMETERIZATION AND MOVE BLOCKING STRATEGIES IN MODEL-PREDICTIVE CONTROL

M.Sc. THESIS

Canan DOMBAYCI
(506111006)

Department of Chemical Engineering

Chemical Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Devrim Barış KAYMAK

DECEMBER 2013

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

MODEL ÖNGÖRÜLÜ KONTROL SİSTEMLERİNDE ADAPTİF KONTROL VEKTÖR PARAMETRİZASYONU VE HAREKET BLOKLAMA

YÜKSEK LİSANS TEZİ

Canan DOMBAYCI
(506111006)

Kimya Mühendisliği Anabilim Dalı

Kimya Mühendisliği Programı

Tez Danışmanı: Assoc. Prof. Dr. Devrim Barış KAYMAK

ARALIK 2013

Canan DOMBAYCI, a M.Sc. student of ITU Graduate School of Science Engineering and Technology 506111006 successfully defended the thesis entitled **"ADAPTIVE CONTROL VECTOR PARAMETERIZATION AND MOVE BLOCKING STRATEGIES IN MODEL-PREDICTIVE CONTROL"**, which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Devrim Barış KAYMAK** Istanbul Technical University	....
Co-advisor :	**Prof. Dr.-Ing. Wolfgang Marquardt** RWTH Aachen University	....
Jury Members :	**Prof. Dr. Yaman Arkun** Koc University	....
	Prof. Dr. İbrahim Eksin Istanbul Technical University	....
	Prof. Dr. Serdar Yaman Istanbul Technical University	....

Date of Submission : **22 November 2013**
Date of Defense : **04 December 2013**

Anneme, Babama ve Kardeşlerime,

FOREWORD

I would like to offer my sincere and grateful thanks to my thesis advisor Assoc. Prof. Dr. Devrim Barış Kaymak for his support, understanding and guidance throughout my graduate and prior studies.

I would also like to thank my co-advisor Prof. Dr.-Ing. Wolfgang Marquardt for giving the chance of studying in Dynamic Optimization group of Process System Engineering Department of RWTH Aachen University in order to continue my thesis. Therefore, I would like to thank to all members of the Process System Engineering Department for their help. Especially, I would like to thank to Dipl.-Ing. Inga Wolf for discussions and her great support.

I would also like to thank my jury members, Prof. Dr. Yaman Arkun, Prof. Dr. İbrahim Eksin and Prof. Dr. Serdar Yaman for critical comments and feedbacks on my thesis.

I am also grateful to Dr. Haluk Arıcan for his support of my efforts and his guidance about life.

Furthermore, I appreciate initial developer of the document preparation system LaTeX and later developers' importance owing to making my life easier.

In addition, I owe my deepest gratitude to Kazım Akar and my beloved friends as they have supported me mentally, cheerfully, motivationally and technically during this thesis study.

Finally, I would like to thank my mother, İkbal Dombaycı, my father, Kamil Dombaycı, my sister, Sema Kirenci, my brother Murak Kirenci and my little sister, Ilgın Dombaycı, for standing behind me, loving me and encouraging through all my life.

December 2013 — Canan DOMBAYCI
(Chemical Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

ATSA	: Adaptive two-stage approach
CSTR	: Continuous stirred-tank reactor
DAE	: Differential-algebraic equations
DIB	: Delta input blocking
DOB	: Delta offset blocking
DyOS	: Dynamic optimization software
IB	: Input blocking
LMPC	: Linear model-predictive control
LTIMPC	: Linear time-invariant model predictive control
MPC	: Model-predictive control
MPT	: Multi-parametric toolbox
NLP	: Nonlinear programming
NMPC	: Nonlinear model-predictive control
NRO	: Number of refining optimization
OB	: Offset blocking
SQP	: Sequential quadratic programming
QP	: Quadratic programming

LIST OF TABLES

LIST OF FIGURES

Page

ADAPTIVE CONTROL VECTOR PARAMETERIZATION AND MOVE BLOCKING STRATEGIES IN MODEL-PREDICTIVE CONTROL

SUMMARY

For the real time implementation of model-predictive control, the computational time to solve the optimal control problem is critical. All the calculations must be finished or terminated within the sampling time of MPC. In order to reduce the computational time, the number of decision variables for the control inputs can be reduced. To this end, adaptation and move blocking strategies can be applied. In this thesis, well-established move blocking strategies for model predictive control are compared with wavelet-based adaptation techniques. All strategies have in common that they strive to reduce the number of decision variables on the MPC horizon. While move-blocking techniques are purely heuristic, adaptation techniques allow for an (almost) optimal discretization of the controls. However, the (almost) optimal discretization of the control move via adaptation techniques may be too fine to actually implement it on actuators and infeasibilities may arise when a coarser and non-optimal grid must be realized. Different techniques to circumvent this problem are investigated.

For these purposes after investigating dynamic optimization, literature search has been done. For the literature search, move blocking is investigated whether there are different strategies rather than input blocking with two number of decision of variables. Then, adaptation techniques for model-predictive control is investigated whether there are other contributions besides the grid refinement method with control vector parameterization. Besides searching move blocking and adaptation strategies, it is investigated that what can be implemented on state of the art actuators and what will be possible in the future? After literature reviews, move blocking and adaptation strategies are compared for identifying criteria to compare move blocking and adaptation techniques and identifying advantages and disadvantages of all strategies. For this identifications, different works are selected according to their difference features.

In addition, a new move blocking strategy Structure Blocking is suggested which considers the dynamics of system and uses move window blocking strategy. Finally, the control performance of important move blocking strategies and adaptation strategy are compared by using two different scenarios and a nonlinear case study is used. First scenario is set-point tracking that tries to track the set-point values of control inputs and controlled output by using regulatory objective function. These set-point values are steady-state values of the case study. Second one is reference trajectory tracking scenario that that tries to track the reference trajectories by using regulatory objective function. Reference trajectories for control inputs and controlled output are optimal trajectories which are calculated by using economic objective function. Also jump in disturbances are implemented both of these scenarios to see the performances. Consequently, the performance analysis has shown that the computational time could

be reduced by reducing the number of degrees of freedom with a little sacrificing the objective value.

Furthermore, when the system's transient behavior has to be considered with reference trajectory tracking rather than set-point tracking, better results are obtained by using the dynamics of the system and shifting the blocking map of the system. Also, the suggested strategy Structure Blocking shows good performance when comparing it to other strategies as regards the objective value and the computational time which is important point for this thesis.

MODEL ÖNGÖRÜLÜ KONTROL SİSTEMLERİNDE ADAPTİF KONTROL VEKTÖR PARAMETRİZASYONU VE HAREKET BLOKLAMA

ÖZET

Bu tezde gelişmiş bir proses kontrol algoritması olan model öngörülü kontrol (model predictive control) sistemleri incelenmiş ve kontrol yönteminin gerçek zamanlı (real time) sistemlere uygulanabilmesi için gerekli olan hesaplama süresinin azaltılmasına yönelik yöntemler araştırılmıştır. Bu hesaplama süresinin azaltılmasında kullanılan hareket bloklama (move blocking) ve adaptasyon yöntemleri konusunda literatür araştırması yapılmış ve bu yöntemler karşılaştırılmıştır.

Model öngörülü kontrol algoritması, gerçek zamanlı sistemlerde uygulandığında hesaplama süresi önem arz etmektedir. Çünkü model öngörülü kontrolör hesaplaması bir hareketli ufuk süresinden diğerine kadar geçen süre içerisinde yani örnekleme zamanı (sampling time) içerisinde bitirilmelidir. Örnekleme zamanı aynı zamanda kontrol girdisinin (manupüle edilen değişkenin) sisteme uygulandığı süredir.

Model öngörülü kontrol algoritmasında, örnekleme zamanı içerisinde birçok karar değişkeni olan bir dinamik optimizasyon problemi çözülmektedir. Karar değişkenleri dinamik optimizasyon probleminin çözüm yöntemine ve model öngörülü kontrol algoritmasının uygulamasına göre kontrol parametreleri veya kesikli zaman noktaları (discrete time points) olabilmektedir. Her hareketli ufuk için çözülen optimizasyon probleminin sonucu olan kontrol girdisinin pratikte uygulanabilir olması ve optimal değere sahip olması için hesaplamanın örnekleme zamanı içerisinde sonuçlanmış olması gerekmektedir. Bu sayede gerçek zamanlı sistemlerde, bir kontrol girdisi sisteme uygulandıktan sonra ardından bir diğer optimal kontrol girdisi sisteme uygulanabilir.

Dinamik optimizasyon problemlerinde hesaplama süresini düşürmek için karar değişkeni sayısını azaltan hareket bloklama ve adaptasyon çalışmaları yapıldığı tespit edilmiştir. Bu yüksek lisans tezinde, model öngörülü kontrol sistemleri için hesaplama süresini azaltmak için kullanılan adaptasyon ve hareket bloklama yöntemleri yapılan literatür araştırması sonucunda karşılaştırılmıştır ve belirlenen stratejilerin performansları ölçülmüştür.

Literatür araştırması sonucunda, hareket bloklama yöntemin,n lineer sistemlerde model öngörülü kontrol algoritması uygulamasında kullanıldığı tespit edilmiştir. Ayrıca hareket bloklama yönteminin, sistemle ilgili tecrübelerden yararlanarak online sistemlere deneyimsel (heuristic) olarak uygulandığı görülmüştür. Böylelikle hareket bloklama yönteminde sistemle ilgili tecrübelerden yararlanılarak karar değişkenleri azaltılmakta ve her kontrol girdisi için dinamik optimizasyon problemi çözülmektedir. Hareket bloklama stratejisinin, model öngörülü kontrol algoritmasında karar değişkenlerini bloklayarak problemin çözümü için gerekli hesaplama süresinin azaltılması olarak tanımlandığı görülmektedir. Bazı çalışmalarda, gelecek hesaplamalarında

karar değişkenlerinin deneysel olarak azaltıldığı tespit edilmiştir. Bir karar değişkeni değerinin, bir önceki veya sonraki karar değişkenine eşit olacağı varsayımı yapılarak karar değişkenleri, bloklama matrisi ile bloklanır. Bazı hareket bloklama teknikleri ise bloklanacak karar değişkenlerine sistem dinamiklerini kullanarak karar vermektedirler. Bazıları ise bloklanan karar değişkenlerini her hareketten sonra kaydırarak hareket eden pencere bloklama (move window blocking) stratejisi ile çözer. Hareket bloklama, dinamik optimizasyon probleminde eşitsizlik kısıtı uygulanmaktadır. Hareket bloklama aynı zamanda gelecekteki lineer çözümleri elimine etme olarak da tanımlanabilir.

Adaptasyon yönteminde ise, kontrol girdisinin yapısı karar değişkenleri eklenerek ya da çıkartılarak belirlenmektedir ve sistem dinamiklerinden yararlanılmaktadır. Adaptasyon yönteminin, lineer olmayan sistemlerde ve hem model öngörülü kontrol algoritması uygulamasında hem de offline dinamik optimizasyon problemlerinin çözümleri sırasında kullanıldığı tespit edilmiştir. Adaptasyon stratejisinde karar değişkenlerinin azaltılması, dalga analizi (wavelet analysis) kullanılarak yapılmaktadır. Adaptasyonda karar değişkenleri birbirlerine göre az ve çok olan iki kesikli sistem karşılaştırılarak yapılır. Dalga analizinden hesaplanan dalga sabitine göre karar değişkeninin eklenmesine veya çıkarılmasına karar verilir.

Hareket bloklama ve adaptasyon yöntemleri arasındaki önemli farklardan birisi, adaptasyon yönteminde karar değişkeninin sistem dinamiklerine göre ekleniyor olmasıdır. Bir diğer önemli fark ise hareket bloklama yönteminde daha çok karar değişkeni ile başlanıp karar değişkenlerinin sayısı azaltılır. Adaptasyon yönteminde ise mümkün olan en az karar değişkeni ile başlanır. Fakat adaptasyon yönteminde dinamik optimizasyon problemi çözüldükten sonra kontrol girdisi değeri sık kesikli aralıklarda değişebilir. Bu sık değişen değerlerin sisteme uygulanması uygun aktüatör seçimi ile olabilmektedir.

Adaptasyon yönteminin, model öngörülü kontrol algoritmasına uygulayabilmek için aktüatörlerin kontrol girdilerini hatasız uygulayabilmeleri gerekmektedir. Ancak kontrol girdilerinin sisteme hatasız olarak uygulanabilmesi konusunda iki farklı görüş bulunmaktadır. Birincisinde, bir model öngörülü kontrol probleminde her kontrol girdisinin örnekleme zamanı içerisinde tek bir sabit değer (piecewise constant) veya lineer fonksiyon (piecewise linear) olarak çözülmesi gerektiği söylenmektedir. Bu nedenle adaptasyon yöntemi uygulanırken kontrol girdisinin sonucunun sabit değer veya lineer fonksiyon olarak çözülmesi için adaptasyon yönteminde çözüm kısıtlanır. İkinci görüşte ise, uygulanabilir bir zaman fonksiyonuna sahip kontrol girdilerinin sisteme uygulanabileceği belirtilmektedir. Günümüzde ise, modülatör aktüatörler kendilerine ait örnekleme zamanı ve kendilerine ait kontrolöre sahip olduğundan eğer kontrol girdisi uygulanabilir bir zaman fonksiyonu ise sisteme uygulamak mümkün olabilmektedir. Bu nedenle bu tip aktüatörleri kullanan sistemlerde kontrol girdisi örnekleme zamanı içerisinde birden farklı değere sahip olabilmektedir.

Bu yüksek lisans tezinin uygulama kısmında önemli hareket bloklama ve adaptasyon yöntemlerinin performansları karşılaştırılmıştır. Bu amaçla bazı hareket bloklama stratejileri, Opto-Econ-Toolbox'a Matlab ortamında eklenmiş ve tezin uygulama kısmında kullanılmıştır. Literatürde performansları karşılaştırılan önemli hareket bloklama stratejilerin yanı sıra, yeni bir yöntem olarak yapı bloklama (structure blocking) stratejisi tezin amacına uygun olarak önerilmiştir. Yapı bloklama stratejisinde dinamik optimizasyon problemi adaptasyon yöntemi yardımı ile çözülmüştür. Ardından dinamik optimizasyon probleminin çözümünden alınan bloklanabilecek karar değişkenleri yapısı, online çözüm için model öngörülü kontrol algoritmasına uygulanmış ve her hareket çözümünde bloklama matrisi hareket eden pencere bloklama stratejisi yardımıyla değiştirilerek çözüme uygulanmıştır.

Yöntemlerin performans analizi için lineer olmayan modele sahip Williams-Otto sürekli karıştırmalı tank reaktörü kullanılmıştır. Bu reaktör modeli kullanılarak seçilen stratejiler iki farklı senaryo ile birlikte incelenmiştir. Seçilen stratejilerin performans analizleri yapılmış ve daha zorlu bir analiz için simulasyon sırasında adım belirsizlikleri (jump in disturbances) uygulanmıştır. Senaryolardaki performans analizinde istenen çıktılar için regülatör objektif fonksiyonu kullanılmıştır. Regülatör objektif fonksiyonu kullanan model öngörülü kontrol algoritmalarında, istenen referans değer ile problemin çözümü sonucunda çıkan çıktı değişkeninin değerleri arasındaki farkın azaltılması amaçlanmaktadır. Birince senaryo kontrol girdilerinin ve kontrol edilen çıktı değerlerinin ayrı ayrı tek bir değeri takip etmesi amaçlanmıştır. Takip edilecek referans değerler, Williams-Otto reaktörünün yatışkın hal değerlerinden alınmıştır. İkinci senaryo ise referans yörüngesi takibidir (reference trajectory tracking) ve zamana bağlı olarak takip edilen referans değerlerin değiştiği takip yöntemi olarak da tanımlanabilmektedir. Bu referans yörünge, ekonomik objektif fonksiyonu kullanılarak elde edilmiş ve reaksiyondan çıkan ürün için maksimum kar hedeflemesi yapılmıştır.

Sonuç olarak performans analizinde uygulanan stratejiler karar değişkenlerini azaltarak hesaplama süresinin azaltılabildiğini göstermiştir. Model öngörülü kontrol algoritmasında hesaplama süresini azaltan stratejilerin performans analizi için uygulanan iki senaryoda da sistemin davranışını göz önüne alarak uygulanan stratejiler iyi sonuç vermişlerdir. Hareket eden pencere bloklama stratejisini uygulayan stratejilerden de iyi sonuçlar elde edilmiştir. Ayrıca yapı bloklama stratejisinin performansı diğer stratejilerle karşılaştırıldığında daha iyi sonuçlar vermiştir.

Özetle bu yüksek lisans tezinde, hareket bloklama ve adaptasyon yöntemlerinin literatür araştırmasının yanı sıra optimal kontrol yörüngesinin sisteme uygulanıp uygulanamayacağı araştırılmış ve uygun görülen stratejiler uygulamaya geçirilerek performans analizleri yapılmıştır.

1. INTRODUCTION

For the real time implementation of model-predictive control (MPC), the computational time to solve the optimal control problem is important. All the calculations must be finished or terminated with in the sampling time of the MPC. In the literature, in order to reduce the computational time, it is common to reduce the number of decision variables for the control inputs. There are two different strategies, adaptation and move blocking, which reduce the computational time. Therefore, the aim of this thesis is searching and comparing the performance of move blocking strategies and adaptation which are used to reduce the computational time (Wolf & Marquardt, 2013).

It is known that move blocking strategies are purely heuristic. On the other hand, there exists some publications on move blocking that introducing different techniques to increase the efficiency of move blocking strategies (Cagienard et al., 2007; Lee et al., 1995). For adaptation, Hartwich et al. (2007) and Hartwich (2013) use a wavelet analysis for deciding the number and location of the grid points which is called grid refinement. The problem tackled by this strategy is to come up with a discretization fine enough to satisfactorily represent the continuous solution. The on-line implementation of adaptation is critical due to the fine discretization because the discretization intervals may be finer that the sampling time. Therefore, different actuators are investigated and the question is answer weather they can realize an optimal control input function within the sampling time.

This thesis consists of seven chapters and is structured as follows. Chapter 2 describes the dynamic optimization problem based on nonlinear process models for the off-line and on-line case. Chapter 3 presents the literature search about the previous works on move blocking and adaptation. Chapter 4 discusses the state of art of actuators which is significant for on-line implementation. Chapter 5 presents the implementation of the move blocking strategies into the linear time-invariant model-predictive controller. Furthermore, the strategies compared in Chapter 6 are explained here. In Chapter 6, the strategies are discussed with two scenarios. Finally, Chapter 7 contains the main results and the conclusions of this thesis.

2. DYNAMIC OPTIMIZATION

In this chapter, dynamic optimization is investigated in detail. First, the nonlinear process model for dynamic optimization is explained. After that, off-line and on-line dynamic optimization are classified. For the off-line optimization, adaptive control vector parameterization is explained which is used by DyOS (Dynamic Optimization Software) as solution method for the sequential approach. DyOS is tailored for dynamic optimization and based on nonlinear models (Assassa et al., 2013). Furthermore, on-line optimization part is explained for OptoEcon-Toolbox which optimizes the nonlinear or linear dynamic optimization problems and solves them as on-line dynamic optimization problems on a moving horizon. Both of the softwares, DyOS and OptoEcon-Toolbox can use gPROMS as modeling languages (Elixmann et al., 2013). Also an introduction for model based optimization of chemical processes on moving horizon is explained here which based on Binder et al. (2001). Besides, the interested reader is referred to Binder et al. (2001) for further explanation in dynamic optimization problems in off-line and on-line applications. .

2.1 Process Model for Dynamic Optimization

In general, process models are either based on first principles such as the conservation of mass, momentum and energy equations or they are based on empirical observation. Besides, process models can also be constructed from a mixture of both as an hybrid case. For a first-principle model, chemical processes can be represented as differential-algebraic equations (DAE):

$$\dot{x}(t) = f(x(t), z(t), u(t), d(t), p, t),\ \forall t \in I, \quad \textbf{(2.1)}$$

$$0 = h(x(t), z(t), u(t), d(t), p, t),\ \forall t \in I, \quad \textbf{(2.2)}$$

$$y(t) = g(x(t), z(t), u(t), d(t), p, t),\ \forall t \in I. \quad \textbf{(2.3)}$$

$x(t) \in R^{n_x}$, $z(t) \in R^{n_z}$ and $y(t) \in R^{n_y}$ denote the differential variable, algebraic variable and output variable vectors, respectively. $u(t) \in R^{n_u}$ represents the vector of manipulated variables for controlling the system, which are also called control inputs. $d(t) \in R^{n_d}$ vector denotes the modeling uncertainties and disturbances. $p \in R^{n_p}$ denotes a vector of time-invariant system parameters. While the function f (**2.1**) describes the differential part of the DAE, the function h (**2.2**) describes the algebraic part of the DAE. In addition, the index of the DAE system is assumed 1 which means the system is an ordinary differential equation system after one differentiation. The time interval $I := [t_0,\ t_f]$, where t_0 and t_f are the starting time and final time, respectively. In addition, the DAE can be solved with the help of a numerical integrator for given consistent conditions ($x(t_0) = x_0$).

2.2 Off-line Dynamic Optimization on a Fixed Horizon

Off-line optimization is described in detail in this section according to the general mathematical problem which is used in DyOS with path and endpoint constraints (Assassa et al., 2013). This dynamic optimization problem is given by following equations:

$$
\begin{aligned}
\min_{u(t)} \quad & \Phi(x,u,y,t_f,p) && \textbf{(2.4)}\\
\textbf{s.t. } \dot{x}(t) &= f(x(t),z(t),u(t),d(t),p,t), \;\; x(t_0) = x_0, && \textbf{(2.5)}\\
0 &= h(x(t),z(t),u(t),d(t),p,t), && \textbf{(2.6)}\\
y(t) &= g(x(t),z(t),u(t),d(t),p,t), && \textbf{(2.7)}\\
lb &\leq x(t),u(t),y(t) \leq ub, && \textbf{(2.8)}\\
lb_f &\leq x(t_f),u(t_f),y(t_f) \leq ub_f. && \textbf{(2.9)}
\end{aligned}
$$

The main idea of solving the dynamic optimization problem (**2.4**) - (**2.9**) is finding the minimum value for the objective function Φ (**2.4**), subject to the DAE (**2.5**)-(**2.7**) and the constraints, i.e. path constraints (**2.8**) and end point time constraints (**2.9**). While the path constraints must be fulfilled for the differential, manipulated or output variables during the interval I, the end point constraints must hold for the final values of these variables within their interval, lower bounds (lb) and upper bounds (ub).

The objective function Φ that is used in the dynamic optimization applications in Equation (**2.4**) can be grouped into two different types. The first type describes economical factors, either to increase the overall profit. The second type of objective is called the regulatory or control objective that tries to minimize the prediction of the outputs and the set point errors while solving control problem. Regulatory objective functions are quadratic functions and have weights for the inputs, R, and the outputs, Q, for the decision of relative importance of input and outputs (van Hessem, 2003).

Optimizing time-invariant system parameters such as the radius of a pipe, the volume of a tank are too general and optimization of time-invariant system parameters p is not considered in this thesis. Although the final time t_f can also be optimized, it is not one of the degrees of freedom to be determined in this master thesis.

2.2.1 Solution methods for off-line dynamic optimization problems

The algorithms which are designed for solving on-line optimization control problems on moving horizon use the algorithms designed for the off-line solution of optimal control problems. For that reason, this section introduces solution strategies of off-line optimization control problems. Though indirect methods also exists, this section solely focuses in the so-called direct methods which are based on a finite dimensional parameterization of the controls and the constraints. These direct methods can be divided into:

- Direct single shooting or sequential optimization,
- Direct collocation or simultaneous optimization,
- Direct multiple shooting.

While direct single shooting (more detailed explanation can be found in Section 2.2.1.1) represents a sequential approach, direct collocation represents a simultaneous approach. Direct multiple shooting may be considered as a hybrid method, as the model equations are valid only on intervals during the solution iterations.

The sequential optimization is also called direct single shooting or late discretization approach. In every iteration step of the optimization method, the optimization problem is solved exactly by a numerical integration method for the current guess of control parameters c. In the sequential technique, system parameters vector c is only degrees of freedom and the optimization problem is restricted for reducing computational complexity (Mitsos, 2012).

The simultaneous solution approach is also called collocation full discretization or early discretization approach. First, the length of the finite elements is chosen. Then, all variables need also to be discretized with respect to the finite elements. After discretization, as approximation of the differential variables in every finite element is calculated with methods such as Runge-Kutta methods, interpolation methods or collocation methods. In the simultaneous solution approach, all discretized quantities (state and control variables) are degrees of freedom for the optimization and the integration does not need to be done in the major iteration of the nonlinear programming (NLP) because of the discretization of the state variables (Mitsos, 2012; Hartwich, 2013).

The direct multiple shooting method can be considered as a hybrid method because of the combination of sequential and simultaneous solution approaches. The direct multiple shooting method performs control vector parameterization like single shooting. But there are more degrees of freedom for the optimization problem which come from dividing the time horizon to subintervals. All these subintervals are solved as sequential optimization problem (Schlegel, 2004; Hartwich, 2013).

2.2.1.1 Direct single shooting

One of the important key in the direct single shooting method is the control vector parameterization. The dynamic optimization problem is transcribed into a NLP by using control parameters c and a time-invariant basis function σ:

$$u_i(t) = \sum_{j=1}^{N_i} c_{i,j}\sigma_{i,j}(t), \quad i = 1, ...n_u. \tag{2.10}$$

$c_{i,j} \in R^{n_u * N_i}$ is the constant control parameters which has different lines for the control input and columns for the basis function. $\sigma_{i,j}(t)$ can be represented by the first order B-splines which represent the piecewise constant control inputs:

$$\sigma_j(t) = \begin{cases} 1, & if \mu_{i,j} \leq \ j \leq \mu_{i,j+1} \\ 0, & else \end{cases}$$

$$j = 1, \ldots N_i,\ i = 1, \ldots, n_u \tag{2.11}$$

The B-splines are located in the center of the chosen knots ($\mu_{i,j}$) (Schlegel, 2004; Hartwich, 2013).

After this information about the control vector parameterization, the steps of the sequential approach can be shown in Figure 2.1 which has 4 main parts: the optimizer, the integrator, the auxiliary tool 1 and the auxiliary tool 2. First, the user defines the control parameters c for the first iteration.

- The objective function, the end point constraints and the path constraints are handled by the optimizer as NLP. The optimizer provides the optimal control parameters (c^*) to the user or the auxiliary tool 1 according to the termination criteria such as optimal $c_{i,j}$. Sequential quadratic programming (SQP) is used as NLP in single shooting which uses a quadratic objective function to find the direction of the local optimum value, and the optimality conditions to determine the step length of the solution. The optimizer needs gradients of the objective and constraints with respect to control parameters $c_{i,j}$, if the optimizer is gradient-based optimizer. Gradients are provided from the auixilary tool 2 and passed to the optimizer.

- In the auxiliary tool 1, the infinite number of degrees of freedom are reduced by fixing the structure of the control variables u(t) with the help of a basis function σ (**2.10**) which is called control vector parameterization. In parameterization, the control variables (u(t)) are represented using series expansion of known functions such as piecewise constant or piecewise linear. These functions called basis functions. Readers who want to have more information about off-line optimization can look at Binder et al. (2001) and Assassa et al. (2013) for detailed explanations.

- Then, the integrator solves the differential equations and determines the control and the state variable values.

- After the integrator has determined the differential variables and passed the control and state variable values to the auxiliary tool 2, the auxiliary tool 2 has all information to calculate the values of the objective, the constraints and their gradients and passes them to the optimizer. In the calculation of the constraints, all of the values are not checked if they are feasible. According to the implementation, only the discretized time points or their center time points will be checked. Moreover, the determination of the gradients are done from so-called sensitivity equations which are differential equations and cause to increase the number of differential equations.

As long as the termination criteria are not fulfilled, steps 1-4 are repeated. If the termination criteria are fulfilled, then the optimal parameters c^* return as the solution of the off-line optimal control problem (Mitsos, 2012).

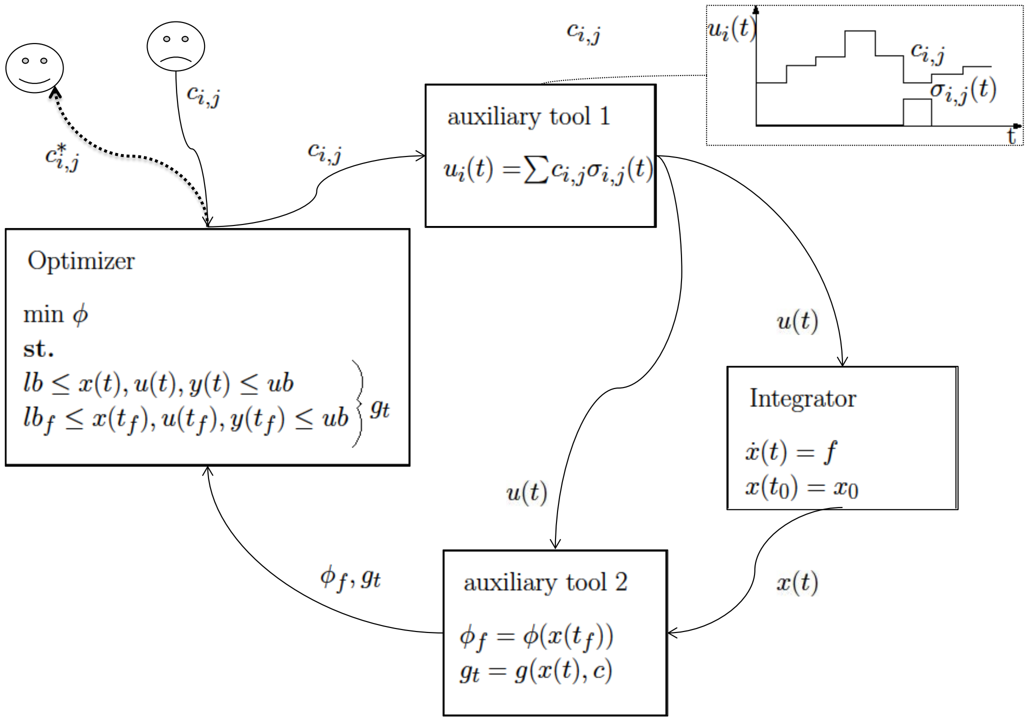

Figure 2.1: Illustration of sequential approach (Mitsos, 2012).

2.2.2 Adaptation and structure detection in off-line dynamic optimization

2.2.2.1 Adaptive control vector parameterization

In the work of Schlegel et al. (2005) and the doctoral thesis of Schlegel (2004), a wavelet-based adaptive refinement algorithm is suggested. Here, the wavelet-based algorithm decides on a suitable discretization of the control inputs. Starting from a coarse grid, grid points are iteratively inserted or eliminated according to the wavelet coefficients values.

In the top line of Figure 2.2, the resolution of the control grid is increased to approximate the dashed curve from S_0 to S_3. In the bottom line of Figure 2.2, wavelet spaces W_i are shown. For W_0, the difference from finer resolution S_1 and coarser resolution S_0 is calculated. For W_1, again the difference from finer resolution S_2 and coarser resolution S_1 is calculated. For W_1, the first two difference are shown as l_1 and l_2 to illustrate where the differences are coming from.

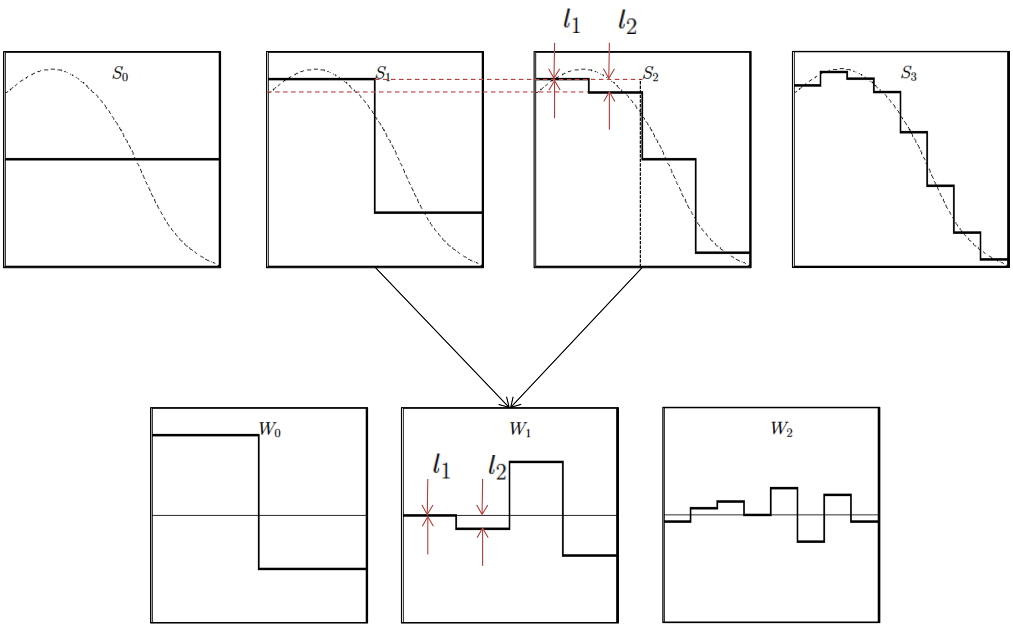

Figure 2.2: Wavelet spaces in adaptive control vector parameterization. Discretization piecewise constant of control in three iteration starting from a coarse grid. Top: space of basis function. Bottom: wavelet space (Schlegel, 2004).

The adaptive refinement method uses the information from wavelet spaces for making decision of inserting or deleting grid points. If the wavelet coefficient values are greater than the defined tolerance, one more grid point is inserted. On the other hand, if the calculated value from the wavelet analysis is below the elimination tolerance, the grid point is eliminated. All the grid points actions are handled as a different case. The reader is referred to Schlegel (2004) and Schlegel et al. (2005) for a more detailed explanation of the adaptive control vector parameterization for dynamic optimization.

2.2.2.2 Structure detection

The idea to apply structure detection originally has been suggested and investigated by Schlegel & Marquardt (2006b). Schlegel & Marquardt (2006b) have developed an algorithm to detect the optimal solution profile. This algorithm reformulates the problem as a multistage problem. This means that the solution of the dynamic optimization problem consists of one or more intervals, so called arcs. Each arc has the same behavior of control action. Four types of arcs can be distinguished:

- u_{min}: the control is at its lower bound,
- u_{max} : the control is at its upper bound,
- u_{path} : the control is determined by an active state path constraint,
- u_{sens} : the control is sensitivity-seeking.

The problem is solved and re-parameterized according to the detected arc structure. Re-parameterization is done to reduce the number of parameters. The fundamental steps of the structure detection are:

- solving the dynamic optimization problem as a single-stage problem and detecting the switching structure according to the arcs,
- re-parameterization and solution as a multistage problem.

2.2.2.3 Adaptive switching structure detection

The adaptive control vector parameterization approach is applied to structure detection by Schlegel & Marquardt (2006a). The aim of using adaptive control vector parameterization and structure detection together is finding (or at least approximately) true continuous time optimal solution. The adaptive switching structure detection is applied because of the coarsely discretized single-stage problem. Because of the coarse discretization, true optimal solution may not coincide with the optimal solution found only by structure detection. Because of this reason, the suggested adaptive switching structure detection first solves the adaptive refinement, which is explained in Section

2.2.2.1. Then the optimal solution is found by detecting the solution as in Section 2.2.2.2.

Approximate structure detection:
For approximate structure detection, Hartwich (2013) adds some simple extensions to the structure detection algorithm which reduces the computational effort. Unnecessary degrees of freedom for each control variable on an intermediate arc u_{path} or u_{sens} is removed. Furthermore, if the length of the arc is below a user defined length, the arc is deleted. For the approximate structure detection, the obtained results are slightly worse than those computed with the original structure detection. On the other hand, the required computational times are favorable for the approximate structure detection.

2.2.3 DyOS

Sequential optimization which is one of the direct method explained in Section 2.2.1, is applied in DyOS (Assassa et al., 2013). General structure of DyOS can be seen in Figure 2.3 (Assassa et al., 2013). If we match Figure 2.1 and Figure 2.3, the optimizer corresponds to the NLP solver in the DyOS, the DAE integrator and auxiliary tool 2 match with the integrator with the help of one modeling language and the auxiliary tool 1 matches with the adaptation block.

In the following DyOS properties and capabilities will be explained for the NLP solver and DAE integrator. In the single-shooting, nonlinear dynamic optimization problem is converted into a NLP problem. This NLP is solved with the help of optimization packages which correspond to the optimizer in Figure 2.1. DyOS uses one of the following optimization packages: NPSOL, SNOPT or IPOPT (Gill et al., 1998; Gill et al., 1996; Wachter & Bieglery, 2006). In this thesis, solely SNOPT is applied.

SNOPT is a general purpose optimizer, designed to find locally optimal solutions for models involving smooth nonlinear functions. SNOPT uses a SQP algorithm. In each major iteration, a quadratic programming (QP) subproblem is solved subject to constraints. Thus, the major iterations try to generate a sequence of solutions for converging to the optimal point, where the first order conditions of optimality are fulfilled. The QP is solved with another software package called LSSOL that gives the minor iteration number for every major iteration. Each new solution c defined by:

$$c' = c + \alpha s \tag{2.12}$$

s is called the search direction and is found by solving the QP. Furthermore, α is the step length (non negative scalar) which is determined for reaching optimality conditions. SNOPT first checks the linear constraints for feasibility and if they are not feasible, SNOPT terminates without checking the nonlinear constraints. If they are feasible, the nonlinear constraints are linearized for the QP (Gill et al., 1996).

DyOS uses LIMEX, NIXE or DDASPK as integrator (Assassa et al., 2013).

In order to integrate the DAE system (as the integrator in Figure 2.1) three different integrator can be used in DyOS. DDASPK , SLIMEX and NIXE correspond to integrator part in Figure 2.1 and Figure 2.3 (Li & Petzold, 1999; Assassa et al., 2013; Schlegel et al., 2004; Hannemann et al., 2010). Since SLIMEX is applied, it will be described in more detail.

LIMEX (v4.2B1) solves a DAE integration problem using an extrapolation method. In the Version 4.2B1, sparse solvers are used for the linear algebra. Within DyOS, SLIMEX a modified version of LIMEX, which was extended by Schlegel et al. (2004), has ability of evaluating sensitivities. Also the auxiliary tool 2 in Figure 2.1 which calculates objective, constraints and gradients is included in the integrator.

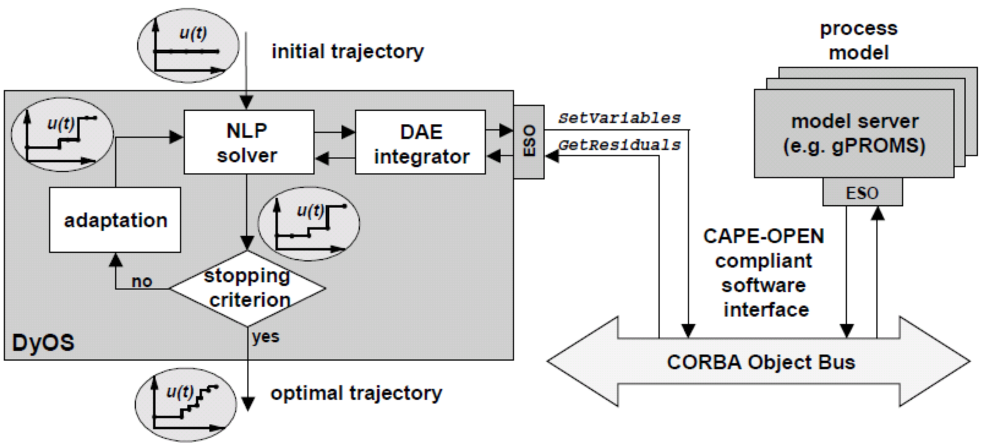

Figure 2.3: Structure of DyOS (Assassa et al., 2013).

2.3 On-line Dynamic Optimization on a Moving Horizon

On-line dynamic optimization on a moving horizon in contrast to off-line dynamic optimization has a moving horizon and error analysis for predicting of model behavior.

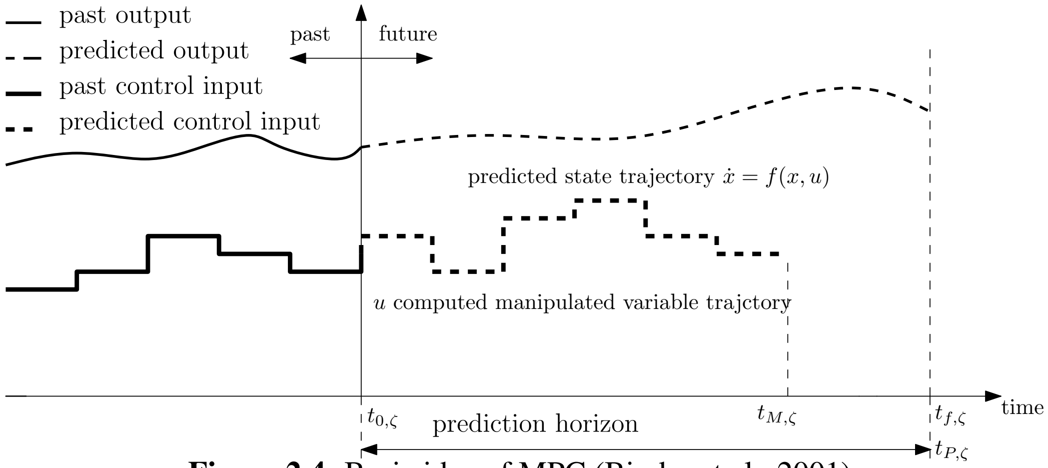

Figure 2.4: Basic idea of MPC (Binder et al., 2001).

On-line optimization on a moving horizon is also called MPC. MPC is an advanced control technique for controlling multi-variable processes. The purpose of this on-line model-based dynamic optimization method, MPC, is to optimize behavior of the process model. The optimization procedure is done with the control inputs by predicting the future behavior of the model outputs. Therefore, the model of the process has a critical role for solving model-based optimization problems (Binder et al., 2001; Rawlings, 2000). As models, which process engineers are dealing with, are nonlinear, many ways are suggested to solve the nonlinear model-predictive control (NMPC) problems. For MPC , the main difficulty is to solve the nonlinear optimization problem on every horizon in real-time.

An illustration of MPC graph is given in Figure 2.4. In figure, $t_{0,\zeta}$ represents the current time of the MPC at the ζ^{th} moving horizon. The prediction horizon length is P and the control horizon is M which are equal in Figure 2.4. Although the prediction horizon and the control horizon calculations are done from time $t_{0,\zeta}$ to $t_{P,\zeta}$ and to $t_{M,\zeta}$, only the control input at time $t_{0,\zeta}$ is implemented into the process. Then, the current time is changed to $\zeta + 1$ and implemented control input and the process output are saved as the past output and the past control input. Hereafter, the predicted control input and the predicted output trajectories are computed again. As it can be seen in Figure 2.4, the control inputs are the manipulated variables and the outputs can be described as states. Mathematically, the optimization problem from (**2.4**) - (**2.9**) is solved on time horizon $I := \left[t_0, t_f\right]$. After that, the control move $u_\zeta(t)$ t $\in I_\zeta$ is applied to process only during the sampling time ΔT. Then, new measurement information is collected, the time horizon are shifted by ΔT and the optimization problem (**2.4**) is resolved on the next time horizon. Every loop for solving the on-line problem can be counted with a horizon index ζ which is also used as subscript in notation, i.e., the horizons is denoted by $I_\zeta := \left[t_{0,\zeta}, t_{f,\zeta}\right]$ (Binder et al., 2001).

Single layer MPC block diagram is shown in Figure 2.5 which considers state feedback and measurable disturbances in the architecture. The process block receives the control input variables which are calculated in the MPC block and disturbances. The MPC block receives the measurable disturbances and the process outputs to calculate the optimal trajectory.

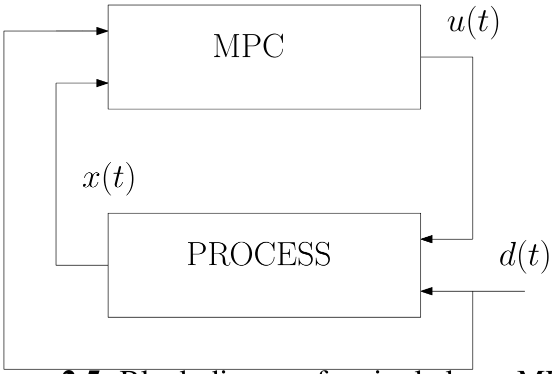

Figure 2.5: Block diagram for single layer MPC.

OptoEcon-Toolbox is used for implementing the MPC environment (Elixmann et al., 2013). OptoEcon-Toolbox is a modular software environment which can use different control blocks for the architecture of the solution. In this thesis, linear time-invariant MPC (LTIMPC) block which is explained in Section 2.3.1 and the Dynopt block which is explained in Section 2.3.2 are used.

2.3.1 Linear time-invariant MPC in OptoEcon-Toolbox

In this thesis, one of the OptoEcon-Toolbox blocks, which uses linear time-invariant model-predictive control (LTIMPC), is extended. The LTIMPC block solves a quadratic regulatory control objective subject to a linearized time-invariant process model and linear constraints. After determining the continuous-time system matrices at the current time from gPROMS, the LTIMPC block translates these system matrices to discrete-time system matrices in Section 2.3.1.1. Then, in Section 2.3.1.2 these discrete-time system matrices are used to build the transition matrices which are transcribed values for the system integration. After that, the transition matrices are used to build the quadratic objective function and the linear constraints for the optimization problem in Section 2.3.1.3. Finally, this transformed problem is solved as QP by using an active set method. The solution strategy for the QP with an active set method is explained in Section 2.3.1.4.

2.3.1.1 Problem formulation of LTIMPC

As the process models are nonlinear, the LTIMPC formulation should consider this. Therefore, the construction of the equations is started from the nonlinear system in the state space representation:

$$\dot{x}(t) = f(x(t), z(t), u(t), d(t), p(t), t), \qquad x(t_0) = x_0, \tag{2.13}$$

$$y = g(x(t), z(t), u(t), d(t), p(t), t). \tag{2.14}$$

State equations (**2.13**) with initial conditions include the differential part (**2.1**) and the algebraic part (**2.2**) of the DAE system. The reason for combining these equations is gPROMS. It gives the differential part and the algebraic part of the system matrices together. Furthermore, (**2.14**) denotes the output of the nonlinear system.

The linearization of (**2.13**) and (**2.14**) is by :

$$\Delta\dot{x}(t) = A^c \Delta x(t) + B^c \Delta u(t) + B^c_d \Delta d(t), \qquad \Delta x(0) = \Delta x_0, \tag{2.15}$$

$$\Delta y(t) = C^c \Delta x(t) + D^c \Delta u(t) + D^c_d \Delta d(t). \tag{2.16}$$

A^c, B^c, C^c, D^c are system matrices which are provided by gPROMS as Jacobian matrices. B^c_d and D^c_d are extracted from Jacobian matrices, when the disturbances are considered as input.

$$\Delta x = x - x_{ss}, \quad \Delta x_0 = x_0 - x_{ss}, \tag{2.17}$$

$$\Delta u = u - u_{ss}, \tag{2.18}$$

$$\Delta d = d - d_{ss}, \tag{2.19}$$

$$\Delta y = y - y_{ss} = y - g(x_{ss}, u_{ss}, d_{ss}). \tag{2.20}$$

Δx, Δu, Δd and Δy (**2.17**) - (**2.20**) represent the difference between the solution trajectories and the steady state values. More detailed explanation of the linearization can be found in Appendix B. Then, this linear continuous time system is discretized with the sampling time of the process (ΔT):

$$\Delta x^{+} \quad = A^{d}\Delta x + B^{d}\Delta u + B_{d}^{d}\Delta d, \quad \Delta x(0) = \Delta x_{0}, \tag{2.21}$$

$$\Delta y \quad = C^{d}\Delta x + D^{d}\Delta u + D_{d}^{d}\Delta d. \tag{2.22}$$

The linear discrete-time system is shown in (**2.21**) and (**2.22**) where the index d represents the discrete-time system matrices. Δx represents the differential variable in discrete-time and $\Delta x^{+} = x(k+1) - x(k)$, where k is discrete-time point. For realizing the integral action, the model is extended by the integral error equation (**2.23**), where y_{ref} is the desired or reference values for the output y.

$$\Delta e^{+} \quad = \Delta e + \Delta t(y - y_{ref}) = \Delta e + \Delta t(\Delta y + y_{ss} - y_{ref}) \tag{2.23}$$

Then, the model equations are extended with the integral action in (**2.24**), (**2.25**) and (**2.26**) which help the LTIMPC block to reach the desired output values.

$$\begin{bmatrix} \Delta x^{+} \\ \Delta e^{+} \end{bmatrix} = \underbrace{\begin{bmatrix} A^{d} & 0 \\ \Delta t C^{d} & I \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \Delta x \\ \Delta e \end{bmatrix}}_{\Delta \hat{x}} + \underbrace{\begin{bmatrix} B^{d} \\ \Delta t D^{d} \end{bmatrix}}_{B} \Delta u + \underbrace{\begin{bmatrix} B_{d}^{d} \\ \Delta t D_{d}^{d} \end{bmatrix}}_{B_{d}} \Delta d + \underbrace{\begin{bmatrix} 0 \\ -\Delta t I \end{bmatrix}}_{B_{y}} (y_{ref} - y_{ss}), \tag{2.24}$$

$$\begin{bmatrix} \Delta x(0) \\ \Delta e(0) \end{bmatrix} \quad = \underbrace{\begin{bmatrix} \Delta x_{0} \\ \Delta e_{0} \end{bmatrix}}_{\Delta \hat{x}_{0}}, \tag{2.25}$$

$$\underbrace{\begin{bmatrix} \Delta y \\ \Delta e \end{bmatrix}}_{\Delta \hat{y}} \quad = \underbrace{\begin{bmatrix} C^{d} & 0 \\ 0 & I \end{bmatrix}}_{C} \begin{bmatrix} \Delta x \\ \Delta e \end{bmatrix} + \underbrace{\begin{bmatrix} D^{d} \\ 0 \end{bmatrix}}_{D} \Delta u + \underbrace{\begin{bmatrix} D_{d}^{d} \\ 0 \end{bmatrix}}_{D_{d}} \Delta d. \tag{2.26}$$

A, B, C, D, B_{d}, B_{y}, D_{d}, $\Delta \hat{x}$, $\Delta \hat{x}_{0}$ and $\Delta \hat{y}$ are used to construct the optimization problem. The constraints are also written in terms of steady state difference:

$$\Delta u_{lb} \quad = u_{lb} - u_{ss}, \tag{2.27}$$

$$\Delta u_{ub} \quad = u_{ub} - u_{ss}, \tag{2.28}$$

$$\Delta y_{lb} \quad = y_{lb} - y_{ss}, \tag{2.29}$$

$$\Delta y_{ub} \quad = y_{ub} - y_{ss}. \tag{2.30}$$

2.3.1.2 Solution of discrete-time system matrices

The transitions matrices are constructed by solving discrete-time linear state equations. Consider the simple example for solving this system (Freeman, 1965).

$$x(k+2) = A^*x(k+1) + B^*u(k+1), \tag{2.31}$$

$$= A^*(A^*x(k) + B^*u(k)) + B^*u(k+1), \tag{2.32}$$

$$= A^{*2}x(k) + A^*B^*u(k) + B^*u(k+1), \tag{2.33}$$

Where A^* and B^* are the system and input matrices of the example. The solution is given by:

$$x(k) = A^{*k}x(0) + \sum_{m=0}^{k-1} A^{*k-m-1}B^*u(m) \tag{2.34}$$

The steady state differences of the outputs samples (Δy, CVs), the input samples (Δu) and the disturbances (Δd) are gathered in the vectors $\Delta\bar{y} \in R^{n_{pred}.n_y}$, $\Delta\bar{u} \in R^{n_{cont}.n_u}$ and $\Delta\bar{d} \in R^{n_{cont}.n_d}$. For the output samples, all prediction horizon samples $\Delta y_{k,i}, with\ k = h, ..., h+n_{pred}-1$ and $i = 1, ..., n_y$, are gathered. For the input samples, all control horizon samples $\Delta u_{k,i},\ k = h, ..., h+n_{cont}-1$ and $i = 1, ..., n_u$, are gathered. For disturbances, all control horizon samples $\Delta d_{k,i},\ k = h, ..., h+n_{cont}-1$ and $i = 1, ..., n_d$, are gathered where k is the current time. For example, if the system has two different inputs ($n_u = 2$), the current time k is equal to 1 and n_{cont} equals to 3, then $\Delta\bar{u}$ is constructed as

$$\Delta\bar{u} = [u_{1,1}; u_{2,1}; u_{1,2}; u_{2,2}; u_{1,3}; u_{2,3}]. \tag{2.35}$$

After defining all input, output and disturbances with all horizon values, the linear time-invariant I/O system:

$$\Delta\bar{y} = T_{[\hat{x}_0 \to y]}\Delta\hat{x}_0 + T_{[u \to y]}\Delta\bar{u} + T_{[d \to y]}\Delta\bar{d} + T_{[y \to y]}\Delta\bar{y} \tag{2.36}$$

More detailed explanations for the construction of the transition matrix can be found in Appendix A.

2.3.1.3 Objective function and constraints of LTIMPC

The LTIMPC has a regularity objective function. The off-line dynamic optimization problem is thus given by:

$$\min_{\Delta\bar{u}} \mathbf{d}\bar{y}' \, Q \, \mathbf{d}\bar{y} + \mathbf{d}\bar{u}' \, R \, \mathbf{d}\bar{u} \tag{2.37}$$

$$\text{s.t.} \quad (\mathbf{2.36}) \tag{2.38}$$

$$\bar{y}_{lb} \leq \bar{y} \leq \bar{y}_{ub}, \tag{2.39}$$

$$\bar{u}_{lb} \leq \bar{u} \leq \bar{u}_{ub}. \tag{2.40}$$

$\mathbf{d}\bar{y}$ and $\mathbf{d}\bar{u}$ represents one of the main ideas of MPC, i.e. they describe the difference between the prediction of future outputs and control inputs and reference set point or reference trajectories.

$$\mathbf{d}\bar{y} = \bar{y} - \bar{y}_{ref} = \Delta\bar{y} + \bar{y}_{ss} - \bar{y}_{ref}, \tag{2.41}$$

$$\mathbf{d}\bar{u} = \bar{u} - \bar{u}_{ref} = \Delta\bar{u} + \bar{u}_{ss} - \bar{u}_{ref}. \tag{2.42}$$

Finally, the objective function (**2.37**) can be transcribed using the transition matrices (from Appendix A), equations **2.41** and **2.42** and the I/O Equation (**2.36**). Moreover, the constraints (**2.39**) and (**2.40**) can also be transcribed with the help of (**2.27**) - (**2.30**) and (**2.36**). The transcribed problem is then:

$$\begin{aligned} \min_{\Delta\bar{u}} \ & \frac{1}{2}\Delta\bar{u}' \left(T'_{[u\to y]} Q T_{[u\to y]} + R\right)\Delta\bar{u} \\ & + \left(\Delta\hat{x}'_0 T'_{[\hat{x}_0\to y]} + \Delta\bar{d}' T'_{[d\to y]} - \bar{y}_{ref} + \bar{y}_{ss}\right) Q \, T_{[u\to y]}\Delta\bar{u} \end{aligned} \tag{2.43}$$

s.t.

$$\Delta\bar{y} = T_{[\hat{x}_0\to y]}\Delta\hat{x}_0 + T_{[u\to y]}\Delta\bar{u} + T_{[d\to y]}\Delta\bar{d} + T_{[y\to y]}\Delta\bar{y} \tag{2.44}$$

$$T_{[u\to y]} \ \geq \ (\bar{y}_{lb} - \bar{y}_{ss}) - T_{[d\to y]}\Delta\bar{d} - T_{[\hat{x}_0\to y]}\Delta\hat{x}_0, \tag{2.45}$$

$$T_{[u\to y]}\Delta\bar{u} \ \geq \ -\left((\bar{y}_{ub} - \bar{y}_{ss}) - T_{[d\to y]}\Delta\bar{d} - T_{[\hat{x}_0\to y]}\Delta\hat{x}_0 \nu\right), \tag{2.46}$$

$$(\bar{u}_{lb} - \bar{u}_{ss}) \ \leq \ I\Delta\bar{u} \ \leq \ (\bar{u}_{ub} - \bar{u}_{ss}). \tag{2.47}$$

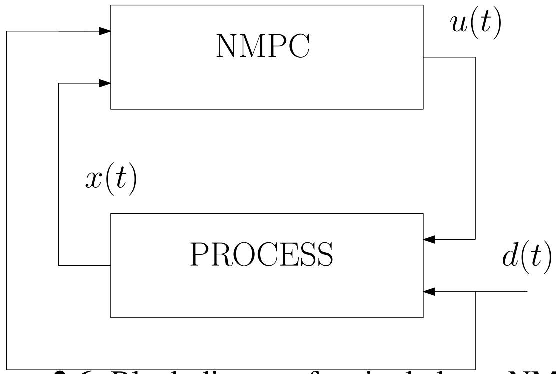

Figure 2.6: Block diagram for single layer NMPC.

2.3.1.4 Solution of QP with active set method

LTIMPC block uses active set methods for solving the convex QP, even if there are many methods such as null space and range-space method. Convex QP is solved after transforming nonlinear continuous system to linear time-invariant discrete optimization problem. In brief, active set method for convex QP states by making a guess of the optimal active set and if this guess turns out to be incorrect, it repeatedly uses gradient and Lagrange multiplier information to drop one index from the current estimate of active set and adds a new index (Nocedal & Wright, 1999). Solving the optimal control problem is done by using *quadprog* function of Matlab (MathWorks, 2013).

2.3.2 Dynopt block in OptoEcon-Toolbox

In this thesis, Dynopt block of the OptoEcon-Toolbox, which uses NMPC, is used for implementing adaptation technique. The Dynopt block solves a NLP by using DyOS which is already explained in Section 2.2.3.

The adaptation method, which is also explained in Section 2.2.2.1, is used as implemented in the OptoEcon-Toolbox Dynopt block which solves an dynamic optimization problem on every moving horizon ζ (Elixmann et al., 2013). On every moving horizon, the initial values of the differential variables are rewritten. The discrete-time intervals for the control inputs are large because a very coarse grid is first defined which is refined while it is tried to find an optimal solution.

Single layer architecture of the NMPC which belongs to Dynopt block is depicted in Figure 2.6. In the on-line case, DyOS is used to solve NMPC problem. Initial values of the differential variables, disturbances as parameters and final time of the moving horizon are given to the DyOS to solve the dynamic optimization problem. Then the optimal control inputs are passed to the process for the simulations and the first control move is applied to the system. For the next moving horizon, all of the information are passed to the Dynopt block which solves the problem with the help of the DyOS until end of the planned horizon.

3. LITERATURE SEARCH

In this chapter, the results of a literature search on move blocking and adaptation are presented. In the real time implementations of on-line dynamic optimization, the computation time of the optimal control problem has a critical role as well as the optimal discretization and number of decision variables of the problem.

Hence, the main purpose of this literature search is analyzing and describing differences and similarities of move blocking and adaptation strategies which have purpose of reducing computational time and complexity In addition to this purpose, finding a new perspective of adjustable technique is aimed with the help of the known literature, which can be implemented to on-line dynamic optimization to compare.

Move blocking and adaptation techniques descriptions are given following. Move blocking is one of methods for reducing the degrees of freedom resulting in the reduction of the computational burden. This reduction is done heuristically, i.e. control inputs are kept constant over several discretization interval which coincide with a sampling time with the help of so-called blocking matrix. Generally, the generation of the blocking matrix or the implementation of blocking procedure is applied without knowledge of the system's dynamics. However, also blocking strategies which consider the system's behavior exist. Moreover, while the move blocking applied for the inputs of the optimization problem, the inequality constraints can also be blocked to avoid feasibility problems. On the contrary, adaptation adapts the control structure, i.e. removes or adds grids for the control according to the dynamics of the system. Furthermore, for this purpose, adaptation use mathematical tools such as wavelet analysis which adapt the trajectory and refining the grid by generally beginning from coarse grids.

This chapter is structured as following. First, the literature search on move blocking is given in Section 3.1. Then, important adaptation techniques are given in Section 3.2 that comprehensive literature search can be found in Schlegel (2004) and Hartwich (2013). After giving the literature search about move blocking and adaptation techniques, literature search is summarized and investigated in the Section 3.3 for categorizing these techniques. The criteria for categorizing move blocking and adaptation strategies are constructed for gathering the similar strategies of move blocking and adaptation methods and identifying the control types of the optimization problem such as the system that they use, the constraints and objective functions. In addition, criteria to distinguish the different move blocking strategies are formulated such as the constructions of the blocking matrix, the implementation of the blocking matrix and solution methods. Finally, some of the move blocking and adaptation strategies are chosen to show the differences and similarity between move blocking and adaptation strategies in Section 3.3.1.

3.1 Literature Search for Move Blocking

In this Section, the literature review about move blocking techniques are given which are used for reducing computational complexity in many MPC application. Although there are a some publications in the literature which discuss move blocking algorithms with MPC, there is an important review for the move blocking which is done by Cagienard et al. (2007). Besides this section is organized for giving the important publications of the groups by observing the chronological order as well as the reference hierarchy between publications. Furthermore, there are some publications which are not directly related with move blocking in on-line dynamic optimization. These publications are not explained, because of mostly using the traditional input blocking which is also explained in Cagienard et al. (2007).

Lee et al. (1995) suggest a method for constructing the blocking matrix, since there was no systematic method for choosing the blocking matrix to reduce on-line computational requirements up to that time. In their paper, blocking is used in linear MPC for linear discrete-time system. The objective function is quadratic and the input and the constraints are implemented in the output of the system. The MPC problem is transformed into the wavelet domain to decide on appropriate choices for the blocking matrix. After the objective function and the constraint equations are written in terms of the wavelet basis which uses haar basis, the blocking matrix is build and implemented for control inputs. According to the article, the blocking matrix can be implemented as an inequality constraints for the input. Besides, the blocking matrix is built from the identity matrix by eliminating the columns corresponding to the elements of the control input. At the same time in this article, condensing is explained as some kind of blocking for the outputs, where insignificant linear combinations of the future output tracking error are eliminated.

Based on Lee et al. (1995), which was submitted in 1992, Palavajjhala et al. (1994) discuss blocking and condensing for QDMC which is a special kind of linear model-predictive control (LMPC). They have applied their suggested strategy for linear discrete-time systems. The control problem comprises a quadratic objective function and input and output constraints are involved. The blocking technique is used for defining the number of samples of the control variables on the prediction horizon. The dynamic matrix A, which contains dynamic behavior of the system and included in the objective function of the QDMC, is reconstructed with two other matrices to reduce the number of samples over the prediction horizon. While Lee et al. (1995) use wavelets for the building blocking matrix by transfering the objective function and the constraint equations into the wavelet domain, Palavajjhala et al. (1994) use wavelets for checking whether the blocking matrix is applicable or not. First, the blocking matrix is constructed in a heuristic way by using system knowledge. Then, the solution for this heuristic blocking matrix is checked by an energy sequence which is provided from a wavelet algorithm. In Palavajjhala et al. (1994), condensing is also discussed which is used for defining the number of the predicted output samples on the prediction horizon. The authors concludes that the predicted output values are more accurate with blocking and design of the blocking matrix can be checked using information which come from wavelet analysis.

Halldorsson et al. (2005) use a nonlinear model for continuous-time systems. The optimal control problem is constructed with a nonlinear objective function and input and state constraints for NMPC. Halldorsson et al. (2005) do not define a blocking matrix. This method uses a different technique than the traditional move blocking strategies for reducing computational complexity: it successively increases the number of discretization intervals by doubling the first sampling interval for the control inputs. In this strategy, the first the control inputs have only two discretization interval, then the first discretization interval of the control input is bisected until desired solution accuracy for the output is reached. Due to the repeated bisection of the first discretization interval, the first discretization intervals are smaller than the intervals which are located far away from the start time of the horizon. This bisection of the first discretization interval can also be considered as an insertion of aa grid point until the desired (optimal) output is reached.

Cagienard et al. (2007) investigate linear discrete-time systems for LMPC with a quadratic objective function. The regulatory optimal control problem is constrained in terms of inputs and states. In the article, input blocking (IB), offset blocking (OB), delta input blocking (DIB), delta offset blocking (DOB) and moving window blocking (MWB) strategies are described, where MWB is suggested as a novel move blocking strategy. Except for MWB, the blocking matrices do not change for different horizons. The MWB matrix, however, changes in every iteration according to a suggested algorithm. All of the blocking matrices are construct heuristically without any knowledge of the system dynamics. The authors use the Multi-Parametric Toolbox (MPT) for the calculations of the blocking schemes (Kvasnica et al., 2004). The results of this work can be listed as follows:

- The Moving window blocking scheme provides feasibility for the constraints and guarantees stability in closed-loop rather than other investigated blocking strategies (IB, OB, etc...).
- Although MWB, OB and DOB sustain feasibility in a small area of the state space around the origin, they are not able to obtain feasible control sequences for states far from the origin in the controllable state space.
- Offset blocking strategies (OB, DOB and MWB) give better performance results for unstable systems than input blocking strategies (IB, DIB).
- IB gives better performance than other blocking schemes for stable systems.

Ricker (1985) uses a blocking strategy for control input variables for an internal model controller. It is suggested to define the manipulated variables for a block of several discretization intervals. For this block, the manipulated variables are held constant. Furthermore, it is suggested to change the control input variable blocking in each calculation and this suggestion is the main idea of the MWB strategy which is suggested by Cagienard et al. (2007).

Gondhalekar et al. (2009) suggest a general approach for LMPC with strong feasibility in linear MPC by implementing a relaxation in the prediction state constraints and Gondhalekar & Imura (2010) extend this relaxation procedure for the prediction state constraints and input constraints. While solving the MPC problem, they directly use the IB and OB strategies which are presented by Cagienard et al. (2007). Gondhalekar & Imura (2010) use the traditional move blocking matrix which has a lower staircase structure and does not change over the MPC moving horizon. On the other hand, Gondhalekar et al. (2009) do not restrict the move blocking matrix as lower staircase or the first element of the blocking matrix (blocking_matrix(1,1)) equals to 1. With the improvement of LMPC algorithm, Gondhalekar & Imura (2010) has better results with using move blocking.

In Olderwurtel et al. (2009) and Oldewurtel (2011), move blocking is used to decrease the computational complexity of affine disturbance feedback MPC. Move blocking is applied for blocking parameterization. Move blocking is only applied to the disturbance feedback matrix because the disturbance feedback matrix has the most number of decision variables and its purpose is to keep the solution of the open-loop prediction problem feasible. For applying move blocking, the vector of decision variables is parameterized by a lower-dimensional vector by combining the vector of decision variables with a blocking matrix. After move blocking parameterization, a least restrictive move-blocking method is used (Gondhalekar & Imura, 2010). It is found that performance of the blocked affine disturbance feedback case is remarkably good and close to the unblocked case.

3.2 Literature Search for Adaptation Techniques

In this Section, the literature review about adaptation techniques are given which mostly used in off-line dynamic optimization rather than on-line dynamic optimization. Although there are many publications in the literature which discuss adaptation techniques, some of the important publications are given for explaining the main idea of the adaptation techniques. Besides this section is organized by observing the chronological order the publications. Furthermore, the interested readers can check Schlegel (2004) and Hartwich (2013) for more detailed literature review.

Betts & Huffman (1998) use nonlinear discrete-time models in the optimal control problem. They use of a quadratic objective function and the nonlinear optimal control problem is subject to input and state constraints. The discretization of control inputs is refined in order to improve the accuracy of the approximation of the optimal control trajectory, i.e. this method refines the mesh of the discrete problem to approximate the continuous one. A grid point refinement method is used, where the discretization error is calculated from two different numerical methods Runge-Kutta and Hermite-Simpson differences to decide weather a new grid point is inserted. The method begins with obtaining the solution of the optimal control problem with the coarse grids. Then, the number and the location of the grid points are decided.

Canto & Banga (2000) propose a mesh refining approach for dynamic optimization. They consider continuous-time nonlinear models with input and state constraints and a quadratic objective function. In the method, second order sensitivities as well as the gradients, and Hessians for the objective function are produced. A mesh refining approach is considered which has serial repeating optimizations with increasing discretization values until the number of refining optimization (NRO) achieved. In the method, number of the initial control discretization intervals, number of the final control discretization intervals and the integration tolerances are set by the user. After the initial control profile is guessed, NRO and tolerance reduction step (r_ρ) are computed with the help of the user settings, then the optimization loop for refining the control discretization is applied. In every loop of the algorithm, discretization number is multiplied with r_ρ to have new discretization value. r_ρ is used to increase the discretization level which is also chosen by user. Authors suggest to use $r_\rho = 2$ for doubling the resolution of the control variables. Then, reduction of integration tolerance and adjustment of the optimization tolerance which are also affects the NRO.

The adaptive control vector parameterization method is first presented in the Ph.D. thesis of Schlegel (2004) and, then, also presented in Schlegel et al. (2005). In this method, nonlinear continuous-time systems are considered. Constraints are formulated for the time independent parameters as well as for the control inputs and states. This method is based on the direct single-shooting approach. Adaptation is applied based on a wavelet analysis. The wavelet analysis is based on comparison of the control parameters from the previous and current time steps. According to the algorithm, inserting a grid point or elimination a grid point is possible accoding to wavelet coefficients. A more detailed explanation for the adaptive control vector parameterization approach can be found in Section 2.2.2.1.

Adaptive two-stage approach (ATSA) for on-line implementation is suggested by Hartwich (2013). ATSA has been applied nonlinear continuous-time models and a quadratic objective function is used. In the suggestion, there are two-stages u_c and u_r which are denoted first and the second stage, respectively. The two-stages are parameterized and equidistant grid points (t_{disc}). Moreover the second stage u_r does not need to have the discretization interval lenghts as the sampling time of MPC. Before starting the algorithm, t_{disc} is chosen greater than the sampling time of on-line optimization (Δt_{opt}), if the Δt_{opt} is small enough in order to safe computational complexity. Number of control parameters and the number of control variables discretization leads to large degrees of freedom. Therefore, ATSA eliminates several control parameters by re-parameterization as the multistage problem as the the switching structure detection which is explained in Section 2.2.2.3. This re-parameterization as a multistage is done after wavelet analysis is performed to delete points which are below to the certain threshold. Close loop formulation of multistage problem is solved in the suggested method. For the first stage (u_c), the control input parameterization is done by piecewise constant. The second stage (u_r) has piecewise linear which lengths are equal to the t_{disc} or its integer multiples because of deleting grid points. Stage duration of the first stage (u_c) is Δt_{opt} for all horizons from the problem formulation and u_c is parameterized only one degrees of freedom. More importantly, the close loop formulation of multistage problem is started with previous horizon time and solution is implemented at current horizon. This means that while the current horizon is implementing, the next horizon stages, control parameters, solution begins to set up. In summary, in the shrinking horizon algorithm after determination of u_c, the scale of the stage u_r is determined. Then the control parameters for both stages are interpolated from the solution. Next, the vector control parameters of the second stage is transformed into the wavelet domain for eliminating grid points which have small coefficient (according to determined scale).

3.3 Summary and Analysis of the Literature Search on Move Blocking and Adaptation

After the literature search on move blocking and adaptation, in this section, the methods are summarized in Table 3.1 and 3.2, respectively. The tables contain information on the system which is considered, the constraints and the objective function to identify the control problem. Moreover the control type and the blocking or adaptation strategies are considered for creating these tables. In the tables, heuristic means that the strategy uses the heuristic knowledge about the system to construct the move blocking. The blocking matrix which is also called blocking map is constructed without using any mathematical tools or dynamics of the system. Furthermore, some of the strategies use the shifted blocking matrix on different horizon and they do not use the same blocking matrix.

3.3.1 Analyzing move blocking and adaptation

A some criteria are determined for the analyze of the move blocking and adaptation strategies:

- system dynamics for building discretization intervals or blocking matrix,
- using system dynamics for checking the suggested discretization intervals or blocking matrix,
- grid refinement, insertion or relocation,
- changing the blocking matrix structure on every horizon
- heuristics.

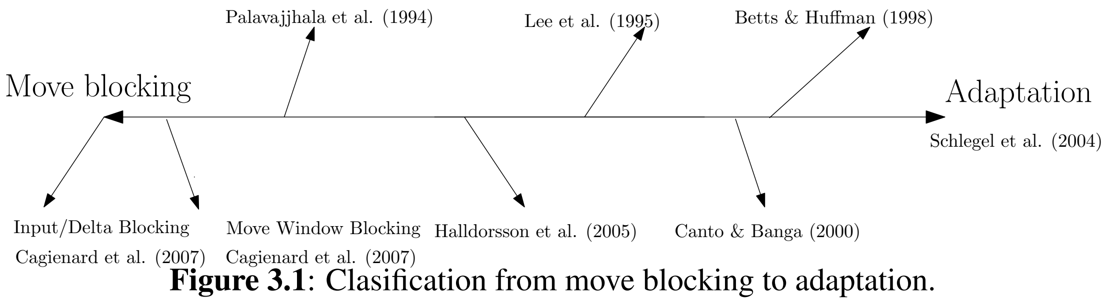

Figure 3.1: Clasification from move blocking to adaptation.

In Figure 3.1, the analyze of strategies are classified from move blocking to adaptation. This classification is done according to how fits the strategies to adaptation or move blocking. The more go to the left side of the line, tells the strategy is closer to move blocking and the opposites tells that the strategy is closer to the adaptation.

When the Input or Delta Blocking of Cagienard et al. (2007) is placed as pure move blocking the MWB of Cagienard et al. (2007) is placed to next because of changing blocking matrix on every moving horizon. Then, the blocking strategy of Palavajjhala et al. (1994) is placed owing to using system dynamics for checking the blocked control inputs discretization intervals. After that Halldorsson et al. (2005) is located in the middle of move blocking and adaptation for using grid refinement, checking the blocked control inputs and using different blocking matrix on every moving horizon. Lee et al. (1995) is located closer to the adaptation because of using system dynamics for constructing blocking matrix and not using heuristics. Canto & Banga (2000) and Betts & Huffman (1998) use system dynamics for building discretization intervals of the control inputs and checks them as well as they do not use heuristics for grid refinement, insertion or relocation. In addition to Canto & Banga (2000) and Betts & Huffman (1998), Schlegel (2004) uses one of the mathematical tool, wavelet analysis on control inputs to decide the grid refinement or insertion.

Table 3.1: Summary of the blocking strategies.

Criteria/Literature	Cagienard et al. (2007) Cagienard et al. (2007)	Palavajjhala et al. (1994) Palavajjhala et al. (1994)	Halldorsson et al. (2005) Halldorsson et al. (2005)	Lee et al. (1995) Lee et al. (1995)
System	Linear discrete-time	Linear discrete-time, input-output model (in obj.), discrete-time	Nonlinear continuous-time system	Linear discrete-time
Constraints	Input and states	Input and output	Input and states	Input and output
Objective function	Quadratic objective function, set-point tracking	Quadratic objective function, trajectory tracking	Nonlinear objective function	Quadratic objective function
Control type	Regulatory LMPC	QDMC (special LMPC)	NMPC	LMPC
Move blocking	no shift, heuristics, blocking types: • input blocking, • offset blocking, • delta input or offset blocking, • moving window blocking	no shift, heuristics but use energy sequence which is provided from wavelet packet algorithm to check the solution if the blocking is not acceptable, a new blocking matrix should suggest	no shift, heuristics (doubled the first control input until desired output is reached), stopping criteria: ε compared to optimal solution	blocking matrix is build after expressing MPC objective function and constraints in terms of wavelet basis

Table 3.2: Summary of the adaptation strategies.

Criteria/Literature	Schlegel (2004) Schlegel (2004)	Canto & Banga (2000) Canto & Banga (2000)	Betts & Huffman (1998) Betts & Huffman (1998)	Rosen & Luus (1992)Rosen & Luus (1992)	Hartwich (2013) Hartwich (2013)
System	Nonlinear continuous-time	Nonlinear continuous-time	Nonlinear discrete-time	Nonlinear continuous-time	Nonlinear continuous-time
Constraints	Input, states and time-invariant parameters	Input and states	Input and states	Input and states	Input, states and time-invariant parameters
Objective function	Quadratic objective function	Quadratic objective function	Quadratic objective function	Quadratic objective function	Quadratic objective function
Control type	Nonlinear optimal control	Nonlinear optimal control	Nonlinear optimal control	Nonlinear optimal control	Nonlinear optimal control
Adaptation	according to performed wavelet analysis, grid points are inserted or eliminated	second order sensitivities are used to calculate gradients and Hessians, number of the discretization intervals are decided from grid refinement	grid refinement method is used and discretization error is calculated to decide if a new grid point should be added	number of discretization interval is doubling until reaching the optimum solution	uses adaptation for NMPC

4. ACTUATORS

In this chapter, the real-time implementation of the optimal control trajectories is investigated based on the capabilities of actuators nowadays. The basic motivation for this part are two articles which contradicting statements. According to Hartwich et al. (2007) and Hartwich (2013), “typical control systems can only employ constant manipulated values over the duration of one sampling interval (“sample and hold”), which does not allow piecewise linear controls or more than one piecewise constant value”. However, according to Pannocchia et al. (2010), “the actuator hardware has become more capable so that any reasonable time function may be sent to the actuator”.

For this reason, possible implementation of different control actions has been investigated for the actuators. This chapter is organized as follows: first general overview of actuators with type is given and the actuators as advanced control instruments are presented.

4.1 Actuators

In the process engineering, actuators are used to apply the force to move the closure of valve. This force needs to overcome the pressure and flow forces which are coming from mechanical design of their parts. The power of this force can be come from different sources and performance of the designed actuators are determined according to these sources (Edgar et al., 2008). Actuators can use three different power as the source, electrical, hydraulic or pneumatic and this definition leads to the three types of actuators (Sclater, 2011).

Pneumatic actuators are the most used actuator types in the industry, even though there are more qualified actuator types. Pneumatic actuators work according to the gas laws and the compressed air is used to move the stem of the actuator. If the accompanying valve is ’push down to close’ the actuator will be described as ’air to close’. A reverse design yields ’air to open’. And this relationship with the compressed air pressure and actuator move has a proportional relationship. The main advantages of these actuators are their high reliability, low cost, sufficient dynamic response, and fail-safe action which are enough properties when comparing to simple design of pneumatic actuators (Edgar et al., 2008).

Hydraulic actuators are designed similar to pneumatic actuators. High pressure is one of the important feature of the hydraulic actuators which is typically 35 to 70 bar. The incompressible nature of the hydraulic oil leads the fast response for transmitting the control actions to the control instruments. The disadvantages are the high initial cost when considering the hydraulic supply and the maintenance (Edgar et al., 2008).

Electric actuators, which are also the main subject of Section 4.2, generally use a typical motor three phase alternative current induction, capacitor-start split-phase induction, or direct current. These devices can provide large thrust, especially given their size (Edgar et al., 2008).

4.2 Actuators as Advanced Control Instruments

Valves can be divided into shut-off valves and modulating valves. *Shut-off valves* are capable of only opening or closing. They will be used in rarely changed process flows because they are only controlled by means of the binary signals open and close. When they are operated, they complete control action when all opening or closing action is complete. On the other hand, modulation valves used to set default variables to constantly verify and adjust the control actions at short intervals. While the control takes place by means of a continuous set point, the time interval between two operations can be a matter of seconds (AUMA, 2013).

Nowadays, the control accuracy of actuators has been improved for using modulation valves and the output speed range for modulating actuators are extended because of an improved mechanical structure, reduced mechanical tolerances and the use of integral controllers. Set-point control can be implemented for valves as the controls receive a position set-point from the host data communication center. The integral positioner compares this value with the current valve position and controls the actuator motor as required until the offset value comes close to zero (AUMA, 2013).

On the other hand, the scan rate of the control module is important and has an influence on the total required CPU processing time of control operation. For this reason, setting the scan rate should be done reasonably. Setting the scan rate is done according to the process dynamics for the analog control modules. When the controlled variable reacts slowly to changes in the control input variable (control temperature), the scan rate of the module should be set slower than when the controlled variable reacts quickly (control of flow rate or liquid pressure) (Emerson, 2013).

Modules that control valves or motors can usually be scanned at 1 second, while the minimum of this scan rate can be at 100 milliseconds. For a 1 second scan rate, outputs are driven soon after the set-point is changed for the module. When undesired interlocks are involved, scan rates can speed up to 500 milliseconds according to Emerson (2013).

This fast scan rates can lead to actuator working condition with quasi-continuous behavior which is explained in Section 4.2.1.

4.2.1 Quasi-continuous actuators

With the technology of the control instrumentation systems, reasonable function for the control action can be implemented without considering the sampling time of the dynamic system. New modules which are implemented in the actuator can reduce the actuators scan rate to 100 milliseconds (Emerson, 2013).

As long as the scan rate of the actuators is much smaller than the discretization interval of the dynamic system, this actuator can be counted as quasi- continuous dynamic system (Weissbach, 2007):

$$\frac{T}{T_{sys}} \ll 1 \tag{4.1}$$

Where T denotes the scan rate of the actuator and T_{sys} donates the sampling time of the dynamic system. Please recall that in dynamic continuous-time systems, the ratio of T and T_{sys} goes to zero.

Furthermore, in Figure 4.1, it can be seen that difference between scan rate of the actuators and the sample time of the dynamic systems. In the figure, there are 3 different control input moves, u_1, u_2 and u_3, with different actuator scan rates, T_1, T_2 and T_3, are represented. The scan rates of the actuators are changed to fit the control input. As shown in figure, while the scan rate is getting finer, the control input move

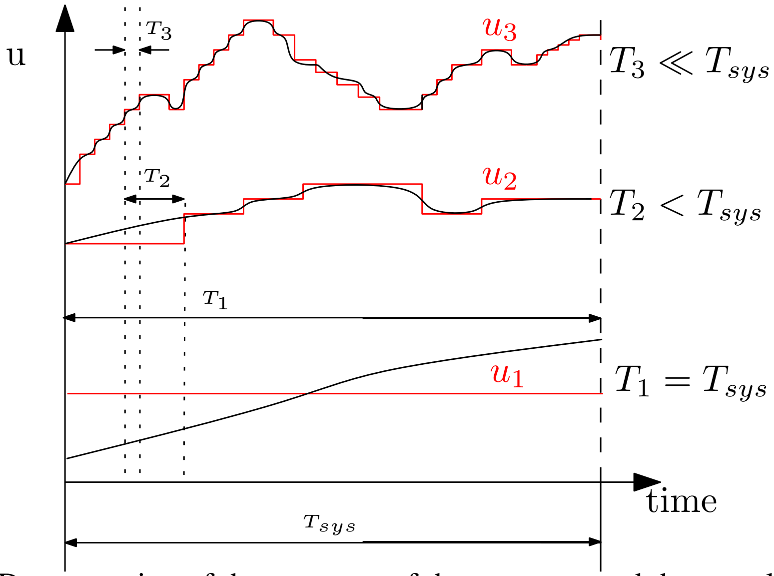

Figure 4.1: Demonsration of the scan rate of the actuators and the sample time of the dynamic system.

is realized better. More importantly, the discretized optimal control input trajectories which are coming from adaptation or move blocking can be realized.

4.2.2 Impact on implementation

In this chapter, the real-time implementation of the optimal control trajectories is investigated based on the capabilities of actuators nowadays and concluded that the state of actuators are capable of implementing the reasonable functions according to their capacity as said by Pannocchia et al. (2010). On the other hand, this control actions will be quasi-continuous according to the definition in Section 4.2.1 which has more than one control inputs in one sample. Modulation actuators are not relevant for move blocking because discretization-time intervals would never be smaller than sampling time. For adaptation, modulation actuators are are important because old actuators restrict refinement depths such that discretization-time intervals are a multiple of sampling time, with new actuators, optimization can continue until optimal value is reached. Please note that in this thesis, the implementation of strategies are done according to piecewise constant and 'sample and hold' control actions which is already implemented in OpteEcon Toolbox.

5. MOVE BLOCKING IN LINEAR TIME-INVARIANT MPC

In this chapter, the implementation of the move blocking schemes for the LTIMPC will be explained as well as the block diagram implemented in the OptoEcon-Toolbox. First, the problem formulation will be explained for the implementation of move blocking. Then, the move blocking strategies implemented including the block diagrams of IB, MWB and structure blocking (SB) are explained for understanding the results which will be compared in Chapter 6.

5.1 Problem Formulation

In Section 2, the following equations for the LTIMPC were defined:

$$\begin{aligned}\min_{\Delta\bar{u}} \ & \frac{1}{2}\Delta\bar{u}'\left(T'_{[u\rightarrow y]}QT_{[u\rightarrow y]}+R\right)\Delta\bar{u} \\ & +\left(\Delta\hat{x}_0'T'_{[\hat{x}_0\rightarrow y]}+\Delta\bar{d}'T'_{[d\rightarrow y]}-\bar{y}_{ref}+\bar{y}_{ss}\right)Q\,T_{[u\rightarrow y]}\Delta\bar{u} \qquad \textbf{(2.43)}\end{aligned}$$

s.t.

$$\Delta\bar{y}=T_{[\hat{x}_0\rightarrow y]}\Delta\hat{x}_0+T_{[u\rightarrow y]}\Delta\bar{u}+T_{[d\rightarrow y]}\Delta\bar{d}+T_{[y\rightarrow y]}\Delta\bar{y} \qquad \textbf{(2.44)}$$

$$T_{[u\rightarrow y]} \ \geq \ (\bar{y}_{lb}-\bar{y}_{ss})-T_{[d\rightarrow y]}\Delta\bar{d}-T_{[\hat{x}_0\rightarrow y]}\Delta\hat{x}_0, \qquad \textbf{(2.45)}$$

$$T_{[u\rightarrow y]}\Delta\bar{u} \ \geq \ -\ \left((\bar{y}_{ub}-\bar{y}_{ss})-T_{[d\rightarrow y]}\Delta\bar{d}-T_{[\hat{x}_0\rightarrow y]}\Delta\hat{x}_0 v\right), \qquad \textbf{(2.46)}$$

$$(\bar{u}_{lb}-\bar{u}_{ss}) \ \leq \ I\Delta\bar{u} \ \leq \ (\bar{u}_{ub}-\bar{u}_{ss}). \qquad \textbf{(2.47)}$$

In the move blocking implementation introduced in this section, the difference from the previous formulation is an equality constraint which will be added for blocking the control inputs with the help of blocking matrix, $T \in R^{n_{cont}\times n_{DOF}}$, and a new control input series $\Delta\widehat{u}$. The equality constraint is given by $\Delta\bar{u}=(TxI)\Delta\widehat{u}$, where I $\in R^{n_{DOF}}$ is the identity matrix and $\Delta\widehat{u} \in R^{n_{DOF}}$ is the blocked control input with a reduced number of decision variables. Hence, $\Delta\widehat{u}$ does not comprise all n_{cont} control input parameters. However, all the control input parameters are still defined implemented in the process block of OptoEcon-Toolbox. In Figure 5.1, the number of discretization intervals are shown for one control input, respectively.

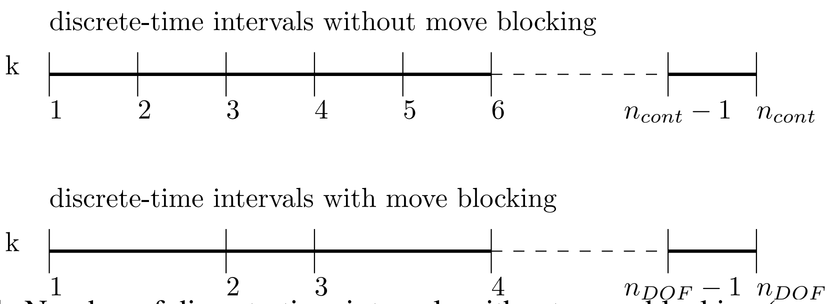

Figure 5.1: Number of discrete-time intervals without move blocking (n_{cont}) and with move blocking (n_{DOF}).

If we omit the moving horizon optimization index ζ for the sake of simplicity, the LTIMPC problem formulation with move blocking is given by:

$$\begin{aligned}\min_{\Delta\widehat{u}} \; & \frac{1}{2}\Delta\widehat{u}'\left(T'_{[u\to y]}QT_{[u\to y]}+R\right)\Delta\widehat{u} \\ & +\left(\Delta x_0'T'_{[x_0\to y]}+\Delta\bar{d}'T'_{[d\to y]}-\bar{y}_{ref}+\bar{y}_{ss}\right)Q\,T_{[u\to y]}\Delta\widehat{u}\end{aligned} \tag{5.1}$$

s.t.

$$T_{[u\to y]}\Delta\widehat{u} \;\geq\; (\bar{y}_{lb}-\bar{y}_{ss})-T_{[d\to y]}\Delta\bar{d}-T_{[x_0\to y]}\Delta x_0, \tag{5.2}$$

$$-(T_{[u\to y]}\Delta\widehat{u}) \;\geq\; -((\bar{y}_{ub}-\bar{y}_{ss})-T_{[d\to y]}\Delta\bar{d}-T_{[x_0\to y]}\Delta x_0), \tag{5.3}$$

$$I\Delta\widehat{u} \;\geq\; (\bar{u}_{lb}-\bar{u}_{ss}), \tag{5.4}$$

$$-\,I\Delta\widehat{u} \;\geq\; -(\bar{u}_{ub}-\bar{u}_{ss}), \tag{5.5}$$

$$\Delta\bar{u} \;=\; (T\otimes I_{n_{DOF}})\Delta\widehat{u}. \tag{5.6}$$

Equations (**5.2**), (**5.3**), (**5.4**) and (**5.5**) represent the inequality constraints and Equation (**5.6**) represents the equality constraint which is applied to implement the move blocking T. In this equality constraint, the move blocking matrix which will either stay the same or change over the moving horizon.

Next, an illustrative example for the move blocking formulation will be given.

5.1.1 Illustrative example for the move blocking formulation

If the control inputs have the same input blocking matrix, then $\Delta\widehat{u}$ could be defined by applying the Kronecker product. The Kronecker product is used for copying the control input values of $\Delta\widehat{u}$ to the blocked control input values $\Delta\bar{u}$ after solving the blocked control problem. If B $\in R^{m_1 \times n_1}$ and $C \in R^{m_2 \times n_2}$, then their Kronecker product B $\otimes$ C will be an $m_1 \times n_1$ block matrix, whose (i, j) block will be the $m_2 \times n_2$ matrix $b_{ij}C$. Therefore, the Kronecker product size will be $(m_1 * m_2) \times (n_1 * n_2)$. An illustrative matrix equation is (Van Loan, 2000):

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} b_{11}c_{11} & b_{11}c_{12} & b_{12}c_{11} & b_{12}c_{12} \\ b_{11}c_{21} & b_{11}c_{22} & b_{12}c_{21} & b_{12}c_{22} \\ b_{21}c_{11} & b_{21}c_{12} & b_{22}c_{11} & b_{22}c_{12} \\ b_{21}c_{21} & b_{21}c_{22} & b_{22}c_{21} & b_{22}c_{22} \end{bmatrix} \tag{5.7}$$

In the following equation, there is an illustrative example for the implementation of the Kronecker product in move blocking. If the LTIMPC problem has two different control inputs, the current time k equals 1, the number of control input samples n_{cont} is 3+3 and, $T = [\ 1\ 0\ ;0\ 1\ ;0\ 1\ ]$, where the new number of degrees of freedom n_{DOF} is 2+2. Consequently, the control input vectors consist of $\Delta\bar{u}$ and $\Delta\widehat{u}$:

$$\begin{bmatrix} \Delta\bar{u}_{1,1} \\ \Delta\bar{u}_{2,1} \\ \Delta\bar{u}_{1,2} \\ \Delta\bar{u}_{2,2} \\ \Delta\bar{u}_{1,3} \\ \Delta\bar{u}_{2,3} \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} \Delta\widehat{u}_{1,1} \\ \Delta\widehat{u}_{2,1} \\ \Delta\widehat{u}_{1,2} \\ \Delta\widehat{u}_{2,2} \end{bmatrix}. \tag{5.8}$$

Thus, we receive:

$$\begin{aligned} \Delta\bar{u}_{1,1} &= \Delta\widehat{u}_{1,1}, \\ \Delta\bar{u}_{2,1} &= \Delta\widehat{u}_{2,1}, \\ \Delta\bar{u}_{1,2} &= \Delta\widehat{u}_{1,2}, \\ \Delta\bar{u}_{2,2} &= \Delta\widehat{u}_{2,2}, \\ \Delta\bar{u}_{1,3} &= \Delta\widehat{u}_{1,2}, \\ \Delta\bar{u}_{2,3} &= \Delta\widehat{u}_{2,2}. \end{aligned}$$

5.1.2 Reconstruction of objective function and constraints

Another important point for implementing the move blocking scheme in Matlab is reconstruction of the objective function and the inequality constraints such that (**5.1**) - (**5.6**). These discretized objective function and inequality constraints can be solved using the Matlab function *quadprog*.

The degrees of freedom should be reduced from $\Delta\bar{u}$ to $\Delta\widehat{u}$. The Hessian matrix of the quadratic objective function and the linear part of the quadratic objective function are needed to be rearranged, while the control inputs are blocked by the blocking map. For the inequality constraints, the important modification is to assign all the inequality constraints of the control input samples to the blocked control inputs after implementing move blocking strategies. As shown in Figure 5.2 for one control input in this example, if the inequalities are not assigned for new sample, there will be infeasibilities. For this reason, when move blocking strategy is implemented, at the same time the lower and upper bounds of the control inputs are copied to blocked control input's lower and upper bound. In other way, the lower and upper bounds do not change after implementing move blocking and finally the blocked control input would have a varying bounds.

5.2 Move Blocking Strategies Implemented in OptoEcon-Toolbox

In this section, some of the move blocking techniques are explained more detailed than in Chapter 3. Input blocking and move window blocking are explained and a new strategy SB is defined in this section. First, input blocking is explained in order to implement pure move blocking technique in Section 5.2.1. Then, the move window blocking strategy is given for explaining shifting move blocking structure in Section 5.2.2. Finally, structure blocking is given in Section 5.2.3 which uses dynamics of the system and shifts the move blocking map at the time.

5.2.1 Input blocking

In the input blocking (IB) method, inputs are fixed to be constant over a certain number of time steps, because the computational complexity of solving the optimization problem on a moving horizon directly depends on the number of degrees of freedom (Cagienard et al., 2007). In order to fix the inputs, blocking matrix T_i is the same on

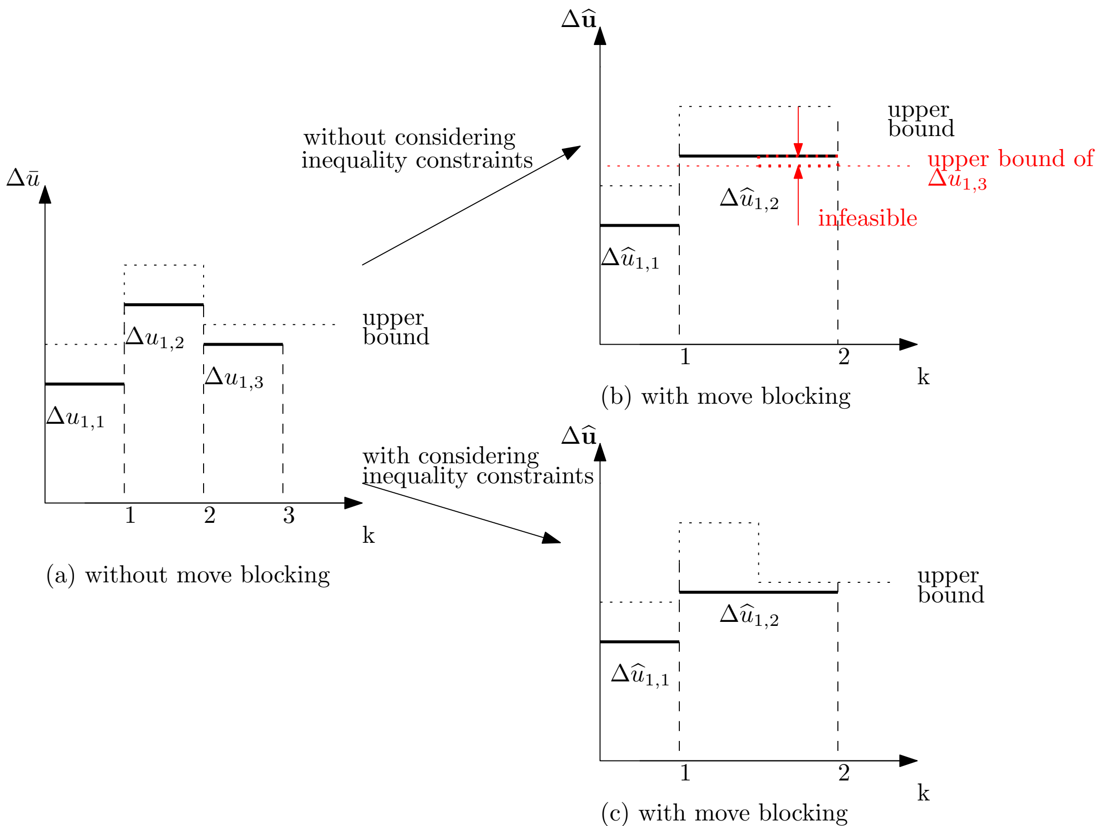

Figure 5.2: Inequality constraints in move blocking formulation which lead to infeasibilities. (a) feasible control move without move blocking for one control input with upper bounds on the control input samples, (b) solution with move blocking but without consideration of the upper bound of $\Delta u_{1,3}$ which may cause infeasibilities, (c) solution with move blocking and with consideration of the upper bounds on $\Delta u_{1,3}$ for $\Delta\widehat{u}_{1,2}$ to prevent infeasibilities.

every moving horizon ζ. In following equation, general equation for move blocking is shown:

$$\Delta\bar{u} \quad = \quad (T \otimes I_{n_{DOF}})\Delta\widehat{u}. \tag{5.9}$$

In the implementation of the LTIMPC block within the OptoEcon-Toolbox, the blocking matrix is defined with the help of a blocking map. For the move blocking matrix $T_{IB} = [\,1\;0\;;0\;1\;;0\;1\,]$,

$$M_{IB} \;=\; \begin{bmatrix} 1 & 2 \end{bmatrix}. \tag{5.10}$$

The block diagram for the IB is depicted in Figure 5.3. The blocks in the block diagram are summarized in the following item:

- *The move blocking options* contain the map information for the LTIMPC.

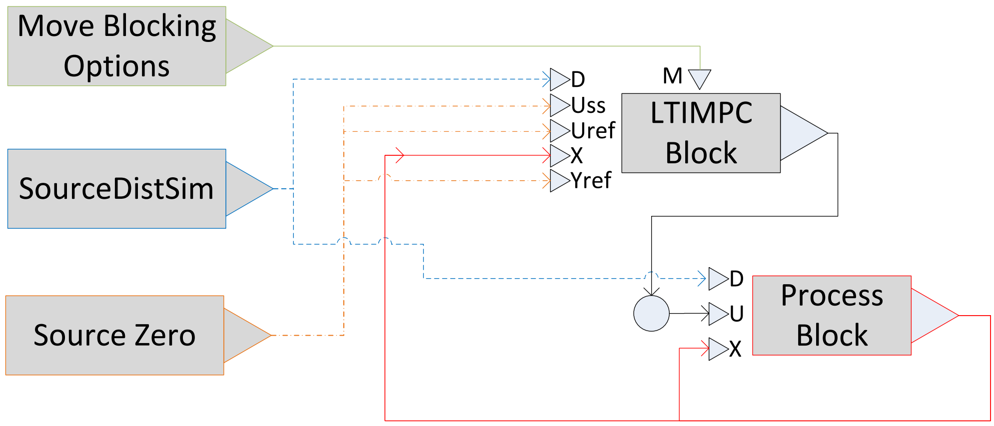

Figure 5.3: Block diagram of LTIMPC with move blocking for IB and MWB.

- *Source zero* contains the initial values for the process, the reference values for the control inputs and the controlled variables (outputs). These values could be only one value for set point tracking or a vector of values for reference trajectory tracking. Furthermore, a control input steady state value (Uss) is required for the set point tracking. For the reference trajectory tracking, the steady state value of the control inputs is coming from the process block because the process will be linearized on every moving horizon.

- *Source disturbance simulation* gives the current the disturbance values to the LTIMPC and the process block.

- *The Process block* uses the current states X, the disturbances D and the control input values U from LTIMPC to calculate the state values for the next sampling time.

- *The LTIMPC block* takes the necessary informations from these blocks and calculates the optimal solution as described in Section 2.3.1.

5.2.2 Move window blocking

In move window blocking (MWB), the move blocking matrix is changed on every horizon (Cagienard et al., 2007). For this purpose, an algorithm which is called blocking matrix generation is created by Cagienard et al. (2007). In this algorithm, first, the admissible upper staircase blocking matrix is defined and the admissible blocking matrix just defined on the first horizon to create the next blocking matrix. Then, the new blocking matrix is implemented while it is used on the next horizon to create new blocking matrix. Then, the last move blocking matrix defined on the previous horizon horizon ζ is used to create the new blocking matrix implemented in (**5.6**). The MWB implementation scheme is depicted in Figure 5.4. In the

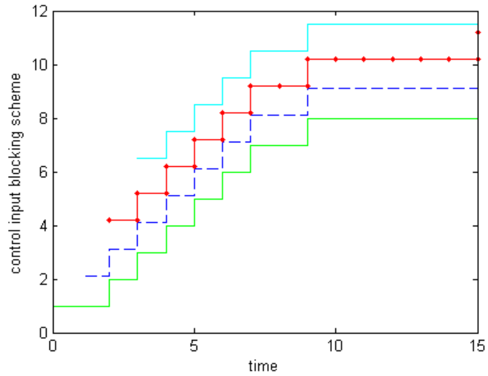

Figure 5.4: MWB scheme for on-line optimization.

implementation of the LTIMPC block, the blocking matrix is defined with the help of a blocking map. The blocking diagram for the MWB is the same as that for the IB (in Figure 5.3). The only difference is the move blocking map configuration.

5.2.3 Structure blocking

In this work, structure blocking (SB) is suggested which uses the adaptation technique explained in Section 2.2.2.1 and constructs the move blocking matrix based on the idea of structure detection in Section 2.2.2.2. In Figure 5.5, the block diagram for SB is shown. Also the algorithm of the SB is presented in Algorithm 1. The difference between the block diagrams for SB, IB and MWB in Figure 5.3 is the Dynopt block which uses adaptation for creating move blocking matrix and implementing the move blocking matrix on every horizon with using MWB.

- *The Dynopt block*, which makes use of adaptation, takes the states X from the process block, initials for the control inputs from source zero and the disturbances from SourceDistSim. Then, it gives the optimized control inputs to the Move Blocking Options for determining the map for the move blocking implementation.

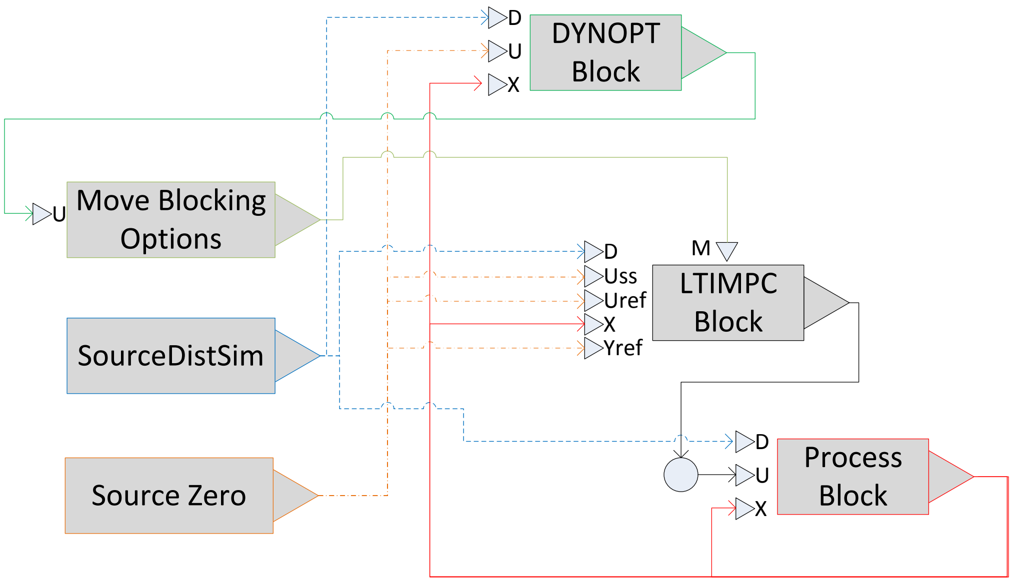

Figure 5.5: Block diagram of LTIMPC with SB.

Algorithm for SB:

Define the number of moving horizon n_ζ and tolerance ε_M for creating map;
Solve the dynamic optimization problem with the Dynopt block as off-line dynamic optimization;
Create the map M_{SB}^1 by checking the sequent discretized control input intervals and block them if they have same value in defined ε_M;
for $\zeta = 1$ **do**
  Translate M_{SB}^1 to T^1;
  Solve blocked LTIMPC dynamic optimization problem with T^1;
  Implement control input for the first horizon;
  $\zeta = 2$;
end
while $\zeta \leq n_\zeta$ **do**
  Use $M_{SB}^{\zeta-1}$ to construct M_{SB}^{ζ} with MWB algorithm;
  Translate M_{SB}^{ζ} to T^{ζ};
  Solve blocked LTIMPC dynamic optimization problem with T^{ζ};
  Implement control input;
  ζ=ζ+1;
end

Algorithm 1: SB Algorithm for move blocking.

6. SIMULATION RESULTS

In this chapter, the control performance of move blocking strategies which are explained in Section 5.2 are compared with adaptation strategy. In order to performance analysis of these strategies, a set-point tracking scenario and a trajectory tracking scenario are considered.

The chapter is structured as follows. First of all, the illustrative case study the Williams-Otto reactor is explained. After the case study explanation, the dynamic optimization system setting up options are defined in Section 6.2. Then, the two scenarios are explained and the results are given and analyzed in Section 6.3.

6.1 Illustrative Case Study

The Williams-Otto continuous stirred-tank reactor (CSTR) which is originally suggested by Williams & Otto (1960) and then used by Forbes (1994) is considered. All methods and scenarios are implemented for the Williams-Otto CSTR example.

6.1.1 Process description

The flowsheet of the Williams-Otto CSTR is shown in Figure 6.1. Three reactions take place in the Williams-Otto CSTR which are exothermic:

$$A + B \xrightarrow{k_1} P \tag{6.1}$$

$$C + B \xrightarrow{k_2} P + E \tag{6.2}$$

$$P + C \xrightarrow{k_3} G \tag{6.3}$$

Products P and E are produced in reactions (**6.1**) and (**6.2**), respectively, and side product G is produced in reaction (**6.3**). All of the reactants are already present in the reactor at the beginning of the simulations as shown in Table 6.1. Reactant A is fed to the reactor with a constant flow rate and reactant B is fed to the reactor continuously with a time-varying flow rate.

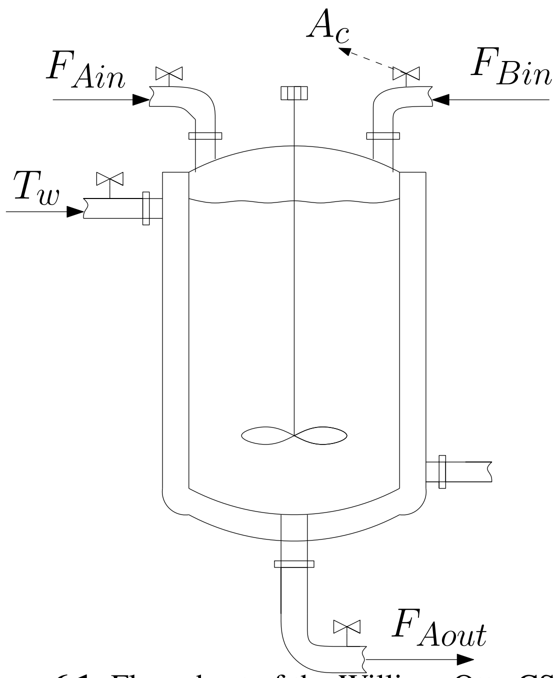

Figure 6.1: Flow sheet of the William-Otto CSTR.

The heat, which is produced by these exothermic reactions, is removed with the help in the cooling water of the cooling jacket. The control inputs for this process are the flow rate of reactant B which is controlled by the control valve cross sectional area A_c and the cooling water temperature T_w.

The model equations for this process are constructed from mass balance equations and energy balance equations. Besides, the constant model parameters which are in these equation are listed and explained in Table 6.2. The differential equations derived from the mass balance are:

$$\frac{dm}{dt} = F_{in} - F_{out}, \tag{6.4}$$

$$\frac{dm_A}{dt} = F_{Ain} - w_A Fout - r_1 m, \tag{6.5}$$

$$\frac{dm_B}{dt} = F_{Bin} - w_B Fout - \left(\frac{M_B}{M_A} r_1 + r_2\right) m, \tag{6.6}$$

$$\frac{dm_C}{dt} = -w_C Fout - \left(\frac{M_C}{M_A} r_1 - \frac{M_C}{M_B} r_2 - r_3\right) m, \tag{6.7}$$

$$\frac{dm_P}{dt} = -w_P Fout - \left(\frac{M_P}{M_B} r_2 - \frac{M_P}{M_C} r_3 r_3\right) m, \tag{6.8}$$

$$\frac{dm_E}{dt} = -w_E Fout - \frac{M_E}{M_B} r_2 m. \tag{6.9}$$

The total mass balance and mass fractions are formulated as:

$$\begin{aligned}
m_i &= \sum_{i=1}^{j} m_i, \forall j \in \{A,B,C,P,E,G\}, && \textbf{(6.10)}\\
F_{in} &= y_{cont} F_{Ain} + F_{Bin}, && \textbf{(6.11)}\\
F_{out} &= y_{cont} F_{in}, && \textbf{(6.12)}\\
w_j &= \frac{m_j}{m}, \forall j \in \{A,B,C,P,E,G\}. && \textbf{(6.13)}
\end{aligned}$$

The reaction constants and rates are:

$$\begin{aligned}
k_1 &= b_1\, exp\left(-\frac{1000\, b2}{T_r + 273.15}\right), && \textbf{(6.14)}\\
k_2 &= b_3\, exp\left(-\frac{1000\, b4}{T_r + 273.15}\right), && \textbf{(6.15)}\\
k_3 &= b_5\, exp\left(-\frac{1000\, b6}{T_r + 273.15}\right), && \textbf{(6.16)}\\
r_1 &= k_1\, w_A\, w_B\, 1000, && \textbf{(6.17)}\\
r_2 &= k_2\, w_B\, w_C\, 1000, && \textbf{(6.18)}\\
r_3 &= k_3\, w_C\, w_P\, 1000. && \textbf{(6.19)}
\end{aligned}$$

The energy balance equations are given by:

$$\begin{aligned}
\frac{dT_r}{dt} &= \frac{J_h + Q + \Delta H_r + \Delta H_v}{m\, C_p}, && \textbf{(6.20)}\\
J_h &= F_{Bin}\, C_p\, T_{Bin} + F_{Ain}\, C_p\, T_{Ain} - F_{out}\, C_p\, T_r, && \textbf{(6.21)}\\
\Delta H_r &= -\left(\Delta H_1\, r_1 + \Delta H_2\, r_2 + \Delta H_3\, r_3\right)\, m, && \textbf{(6.22)}\\
\Delta H_v &= -\left(F_{in} - F_{out}\right)\, C_p\, T_r, && \textbf{(6.23)}\\
Q &= -\frac{A_{ref}\, V\, U}{V_{ref}} \left(T_r - T_w\right), && \textbf{(6.24)}\\
m &= \rho V. && \textbf{(6.25)}
\end{aligned}$$

The flow rate of reactant B, F_{Bin}, is defined with the valve equation:

$$F_{Bin} = C_v\, A_c\, \sqrt{2\, \rho\, \Delta P} \qquad \textbf{(6.26)}$$

where C_v is the valve coefficient and a disturbance. This equation is important because of having control input and disturbances at the same time.

Table 6.1: Initial conditions and initial weight fractions of reactants in the Williams-Otto continuous stirred-tank reactor.

Reactant	Initial Value	Unit	Weight Fraction	Initial Value
m_A	193.3	kg	w_A	0.0962
m_B	777.0	kg	w_B	0.3866
m_C	33.6	kg	w_C	0.0167
m_P	202.5	kg	w_P	0.2866
m_E	576.0	kg	w_E	0.1007
m	2010.0	kg	w_G	0.1132
T_r	80	^oC		

6.1.2 Optimal control problems

Two different objective functions have been used in this thesis. The first one is the regulatory objective function for the performance analysis. The regulatory objective function which is also a quadratic objective function is given in (**2.43**) and (**5.1**) for without move blocking and for move blocking strategies, respectively.

The second objective function is the economic objective function, which is only used in reference trajectory tracking scenario, for maximizing the amount of the products P and E at the same time the profit of this process. The economic objective function is given as following equation:

$$\min \quad \Phi_{eco}, \tag{6.27}$$

where

$$\Phi_{eco} = -1143.38 \quad w_p\, F_{out} - 25.92\, w_e\, F_{out} + 114.34\, F_{B_{in}} + 76.23\, F_{A_{in}}. \tag{6.28}$$

Φ_{eco} represents the profit of the simulations that should be maximized and this economic objective function is only used for producing optimal reference trajectory of the reference trajectory tracking scenario. Other than producing reference trajectory for reference trajectory tracking scenarios, regulatory objective function is used.

Table 6.2: Parameters of the Williams-Otto continuous stirred-tank reactor.

Parameter	Value	Unit	Explanation
b_1	$1.6599 * 10^6$	$1/s$	Arhenius constant
b_2	6.6667	$^\circ K$	Arhenius constant
b_3	$7.2117 * 10^8$	$1/s$	Arhenius constant
b_4	8.3333	$^\circ K$	Arhenius constant
b_5	$2.6745 * 10^{12}$	$1/s$	Arhenius constant
b_6	11.111	$^\circ K$	Arhenius constant
M_A	100	kg/mol	molecular weight of comp. A
M_B	100	kg/mol	molecular weight of comp. B
M_C	200	kg/mol	molecular weight of comp. C
M_P	100	kg/mol	molecular weight of comp. P
M_E	200	kg/mol	molecular weight of comp. E
M_G	200	kg/mol	molecular weight of comp. E
C_P	4.184	kJ/kg	specific heat capacity
ρ	1000	kg/m^3	density of mixture
ΔH_1	-263.8	kJ/kg	reaction heat
ΔH_2	-158.3	kJ/kg	reaction heat
ΔH_3	-226.3	kJ/kg	reaction heat
A_{ref}	9.29034	m^2	heat exchanger area
V_{ref}	2.1052	m^3	heat exchanger volume
U	20	$kJ/(m^2\ K\ s)$	heat transfer coefficient
C_v	0.9	$-$	valve coefficient
ΔP	50000	Pa	valve pressure different

During all dynamic optimization problems with regulatory objective function and economic objective function, the upper and the lower bounds of the valve cross section area A_c, the temperature of the cooling jacket T_w and the temperature of the reactor T_r are assigned as path constraints. These constraints are:

$$0\,cm^2 \leq A_c(t) \leq 11.1\,cm^2, \quad \textbf{(6.29)}$$

$$20^\circ C \leq T_w(t) \leq 100^\circ C, \quad \textbf{(6.30)}$$

$$60^\circ C \leq T_r(t) \leq 90^\circ C. \quad \textbf{(6.31)}$$

6.2 Set-up DRTO System for LTIMPC and Dynopt

Two different OptoEcon-Toolbox blocks are used for the implementations. The first one is the LTIMPC block in which the move blocking strategies are implemented. The second block is the Dynopt block which makes use of the adaptation technique. This block is also used for the strategy suggested in Section 5.2.3 which uses move blocking. There are three different configurations of the two blocks considered in this thesis. The first one is the LTIMPC configuration for the blocking matrix. The second one is the Dynopt block configuration for a regulatory objective function. Finally, the third one is the Dynopt block configuration for an economic objective function. The configuration of the LTIMCP and the Dynopt blocks are given in Appendix C and D and are explained in Sections C and D, respectively.

6.3 Scenarios for On-line Optimization

In this section, all information on the results is gathered for two different scenarios. The first scenario is the set-point tracking described in Section 6.3.1 and the second scenario is the reference trajectory tracking described in Section 6.3.2.

In these two scenarios, the implemented disturbances are the same as shown in Figure 6.2. However, reference trajectory tracking scenario is analyzed with two cases that one of the cases is without disturbances and the other one is with disturbances. The disturbances are the temperature of reactant B, T_{Bin}, and the valve coefficient C_v which has an influence on the flow rate of B reactant.

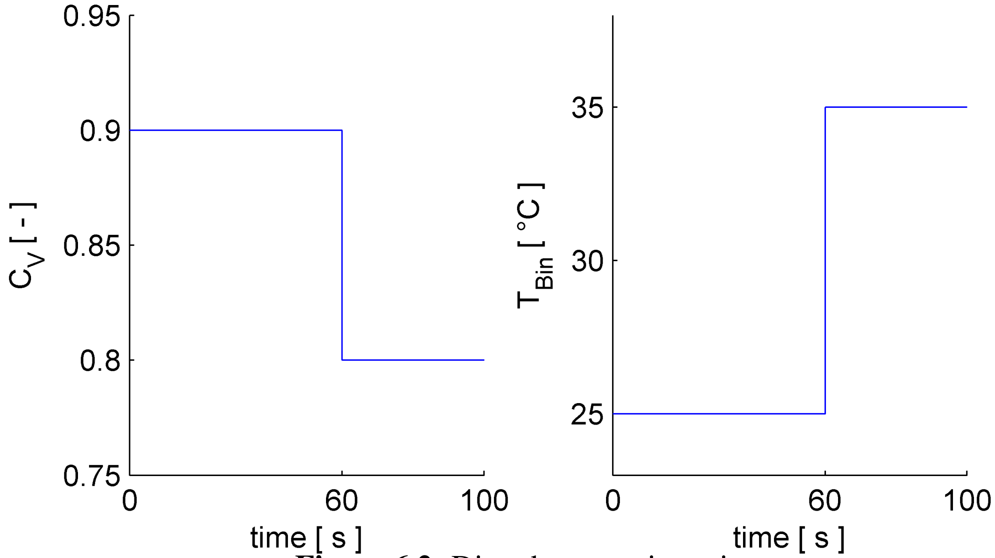

Figure 6.2: Disturbance trajectories.

As shown in Figure 6.2, both the disturbances are constant until $t = 60\ s$. Then, a disturbance jump occurs at time 60 s. T_{Bin} increases from 25^oC to 35^oC and C_v decreases from 0.9 to 0.8 until the end of the simulation.

In the set-point tracking scenario described in Section 6.3.1, a regulatory objective function is considered which tries to settle down the steady state values for the control inputs and controlled variables. In the reference trajectory tracking scenario described in Section 6.3.2, the regulatory objective function is also applied which tries to track the optimal reference trajectory produced with the help of the economic objective function.

The sampling time of the system and the sampling time of the MPC are set to 10 s. The number of cycles is chosen to 25.

6.3.1 Set-point tracking performance

In this section, the performance of the adaptation, the LTIMPC, and the LTIMPC with move blocking strategies are discussed for set-point tracking. For the set-point tracking, the optimal steady state values of the process are determined by using DyOS (Assassa et al., 2013). Then, the performance is investigated with the help of different move blocking strategies and the adaptation technique. The summary and the explanations of the compared strategies are given in Table 6.3. For IB_2 and MWB_2, the blocking maps are decided in a heuristic way from optimal solution of LTIMPC. In addition, adaptation does not use a blocking matrix and begins from a very coarse grid due to the nature of the grid refinement method.

Table 6.3: Strategies and explanations of the abbreviations for the results of set-point tracking.

Abbreviation	Type	Blocking map
LTIMPC	LTIMPC without blocking	none
ADAPT	Adaptation	none
IB_1	Input blocking	$M_{IB} = [1\ 15]$
MWB_1	Move window blocking	$M_{MWB} = [1\ 15]$
IB_2	Input blocking with different maps for the control inputs	$M_{A_c,MWB} = [2\ 1\ 1\ 1\ 1\ 1\ 2\ 6\ 1]$, $M_{T_w,MWB} = [2\ 1\ 1\ 1\ 2\ 8\ 1]$
MWB_2	Move window blocking with different maps for the control inputs	$M_{A_c,MWB} = [2\ 1\ 1\ 1\ 1\ 1\ 2\ 6\ 1]$, $M_{T_w,MWB} = [2\ 1\ 1\ 1\ 2\ 8\ 1]$
SB	Structure blocking: the maps are extracted from Dynopt block but at the same time MWB is implemented	$M_{A_c,MWB} = [3\ 1\ 12]$, $M_{T_w,MWB} = [3\ 1\ 12]$

In Figure 6.3, trajectories of the controlled output T_r (6.3a), the weight fraction w_p (6.3b), the control input A_c (6.3c) and the control input T_w (6.3d) are shown for every strategy listed in Table 6.3.

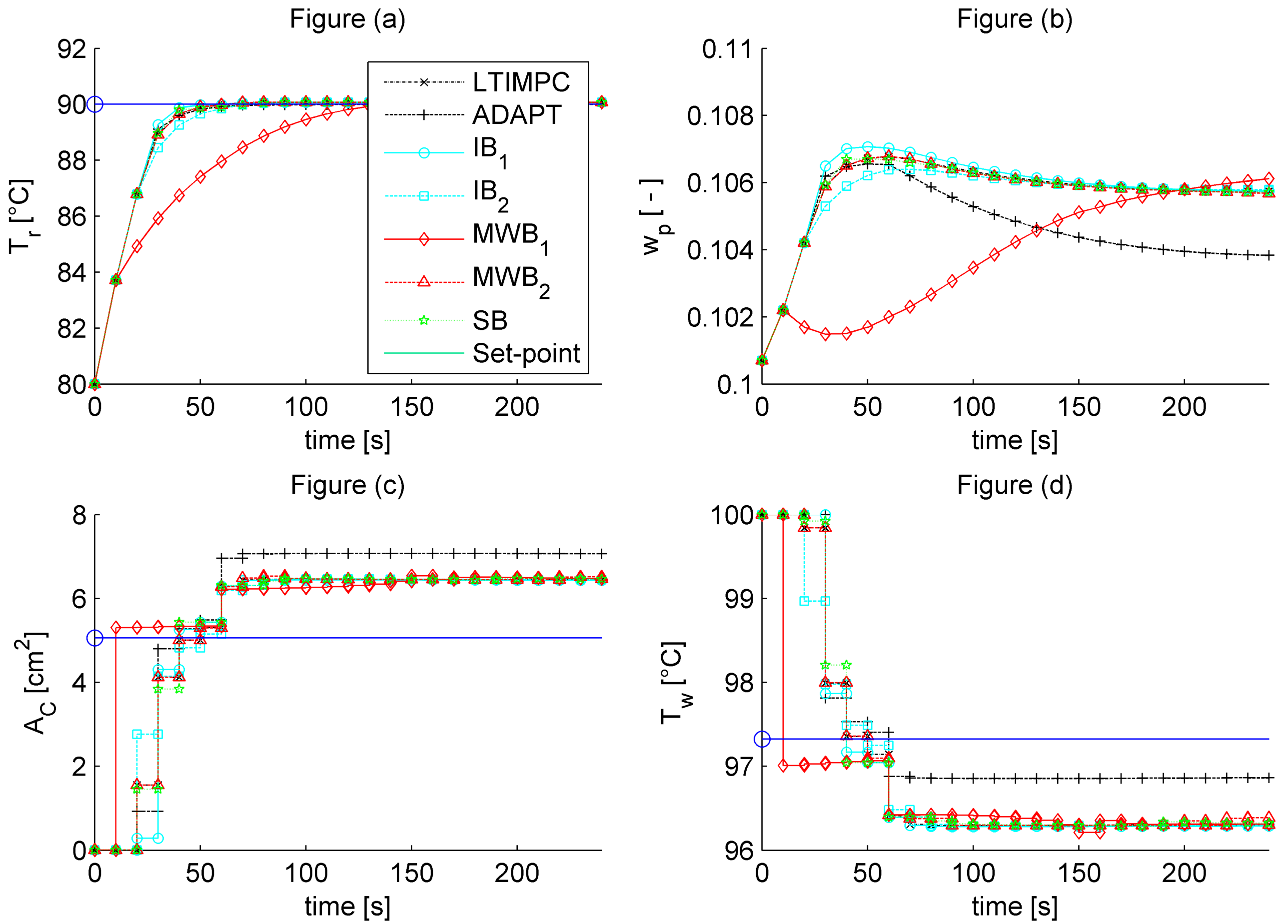

Figure 6.3: Results of all strategies for set-point tracking performance scenario. (a) Process output for the controlled variable, (b) Weight fraction w_p, (c) Control input A_c, (d) Control input T_w.

In Table 6.4, the results for the simulations are given. In the first column, the abbreviations of the strategies are given. The second column of the table comprises the reference objective values compared to the relative objective value for the LTIMPC in percentage. The control objective function values are given at time 250 s. These objective values are computed without delay. The third column of Table 6.4 contains the number of degrees of freedom n_{DOF}. n_{DOF} of ADAPT represents only the first horizon. After the first horizon, these numbers change because of the adaptation strategy. In the table, n_{DOF} is given for both control inputs, e.g. if n_{DOF} is '9+7', it means that the first control input $F_{B_{in}}$ has 9 degrees of freedom and the second control input T_w has 7 degrees of freedom. The last column is the CPU time which shows the calculation time for solving the NLP from (**5.1**) with the Matlab function *quadprog*. ADAPT does not have this CPU time value because the *quadprog* function is not used.

Table 6.4: Numerical results of the strategies for set-point tracking.

Abbreviation	Objective (%)	n_{DOF}	CPU time (s)
LTIMPC	100	16+16	0.0263
ADAPT	99.15	6+5	n/a
IB_1	100.19	2+2	0.0030
MWB_1	150.24	2+2	0.0037
IB_2	100.14	9+7	0.0097
MWB_2	100.09	9+7	0.0093
SB	100.09	3+3	0.0043

6.3.1.1 Discussion of results for set-point tracking

As expected, the results in Table 6.4 and Figure 6.3 show that the CPU reduces when reducing n_{DOF}. *ADAPT* achieves the smallest and thus the best objective function value which means *ADAPT* perform better than other strategies. MWB_2, *SB* and *LTIMPC* have almost the same objective function value, even though *SB* and MWB_2 have less degrees of freedom. It is important to note that *SB* reduces the CPU time without heuristics and has a better performance than MWB_2 because *SB* has less n_{DOF}. MWB_1 has the worst performance of all strategies and even though it has the same n_{DOF} as IB_1. IB_2 which has more n_{DOF} than IB_1 does not perform significantly better. In summary, the strategies where the horizon is shifted show better control performance if they have enough n_{DOF}. Furthermore, it can be concluded that good control performance cannot be guaranteed if heuristics are used.

6.3.2 Reference trajectory tracking performance

Before analyzing the performance of the adaptation and move blocking strategies in the reference trajectory tracking scenario, all strategies are tested without disturbances, in Section 6.3.2.1. In Section 6.3.2.2, the reference trajectory tracking scenario is considered with disturbances. Both cases use optimal reference trajectories for the control inputs and the controlled output for tracking. For both cases with disturbances and without disturbances, same configurations are used. In the reference trajectory tracking scenario, the configurations are set as for the set-point tracking scenario. However, the reference tracking trajectory of the controlled output T_r and the control inputs trajectories are not constant. The reference trajectories are taken from the Dynopt block, where they are computed by an off-line dynamic optimization with the economic objective function. The reference trajectories, which are provided by the Dynopt block, are shown in Figure 6.4.

6.3.2.1 Reference trajectory tracking without disturbances

In Table 6.6, the results of used strategies are summarized for the reference tracking scenario without disturbances. Strategies and explanations of the abbreviations for the reference tracking without disturbances are given in Table 6.5.

In Figure 6.5, the trajectories of the controlled output T_r (6.5a), the weight fraction w_p (6.5b), the control input A_c (6.5c) and the control input T_w (6.6d) are shown for every strategy listed in Table 6.5. In Table 6.6, the results for the simulations are given.

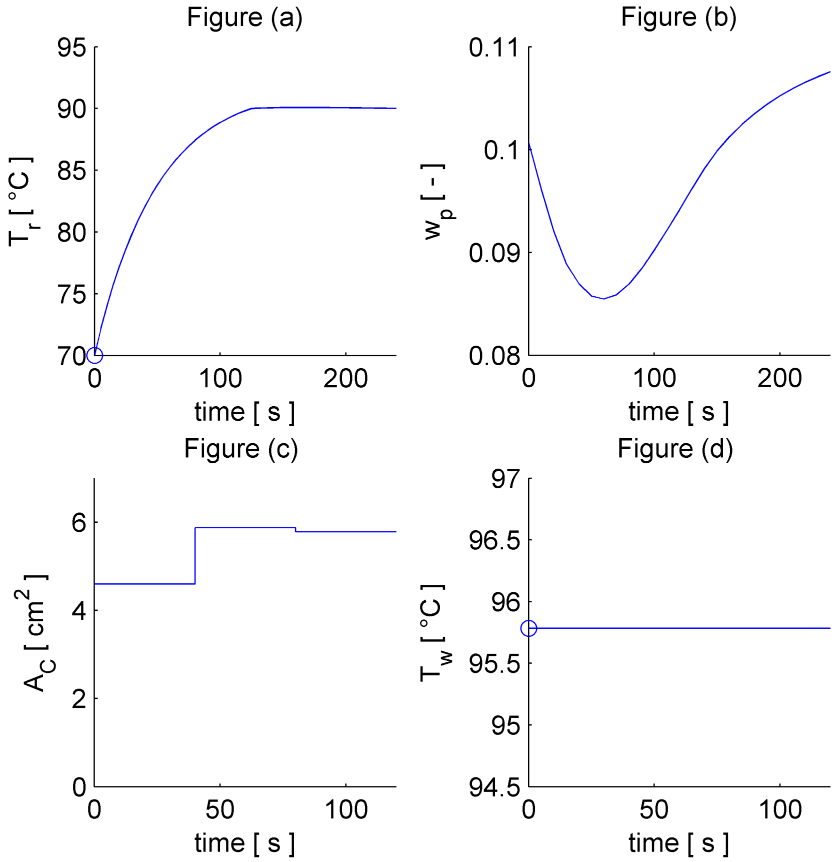

Figure 6.4: Reference trajectories of the reference trajectory tracking scenario.

Table 6.5: Strategies and explanations of the abbreviations for the reference tracking results.

Abbreviation	Type	Blocking map
LTIMPC	LTIMPC without blocking	none
ADAPT	Adaptive Parameter Refinement Method	none
IB_1	Input blocking	$M_{IB} = [1\ 15]$
MWB_1	Move window blocking	$M_{MWB} = [1\ 15]$
IB_2	Input blocking with different maps for the control inputs	$M_{A_c,MWB} = [1\ 1\ 1\ 2\ 2\ 2\ 1\ 4\ 1\ 1]$, $M_{T_w,MWB} = [1\ 2\ 4\ 3\ 2\ 1\ 1\ 1\ 1]$
MWB_2	Move window blocking with different maps for the control inputs	$M_{A_c,MWB} = [1\ 1\ 1\ 2\ 2\ 2\ 1\ 4\ 1\ 1]$, $M_{T_w,MWB} = [1\ 2\ 4\ 3\ 2\ 1\ 1\ 1\ 1]$
SB	Structure blocking: the maps are extracted from Dynopt block but at the same time MWB is implemented	$M_{A_c,MWB} = [1\ 1\ 1\ 1\ 1\ 2\ 6\ 3]$, $M_{T_w,MWB} = [16]$

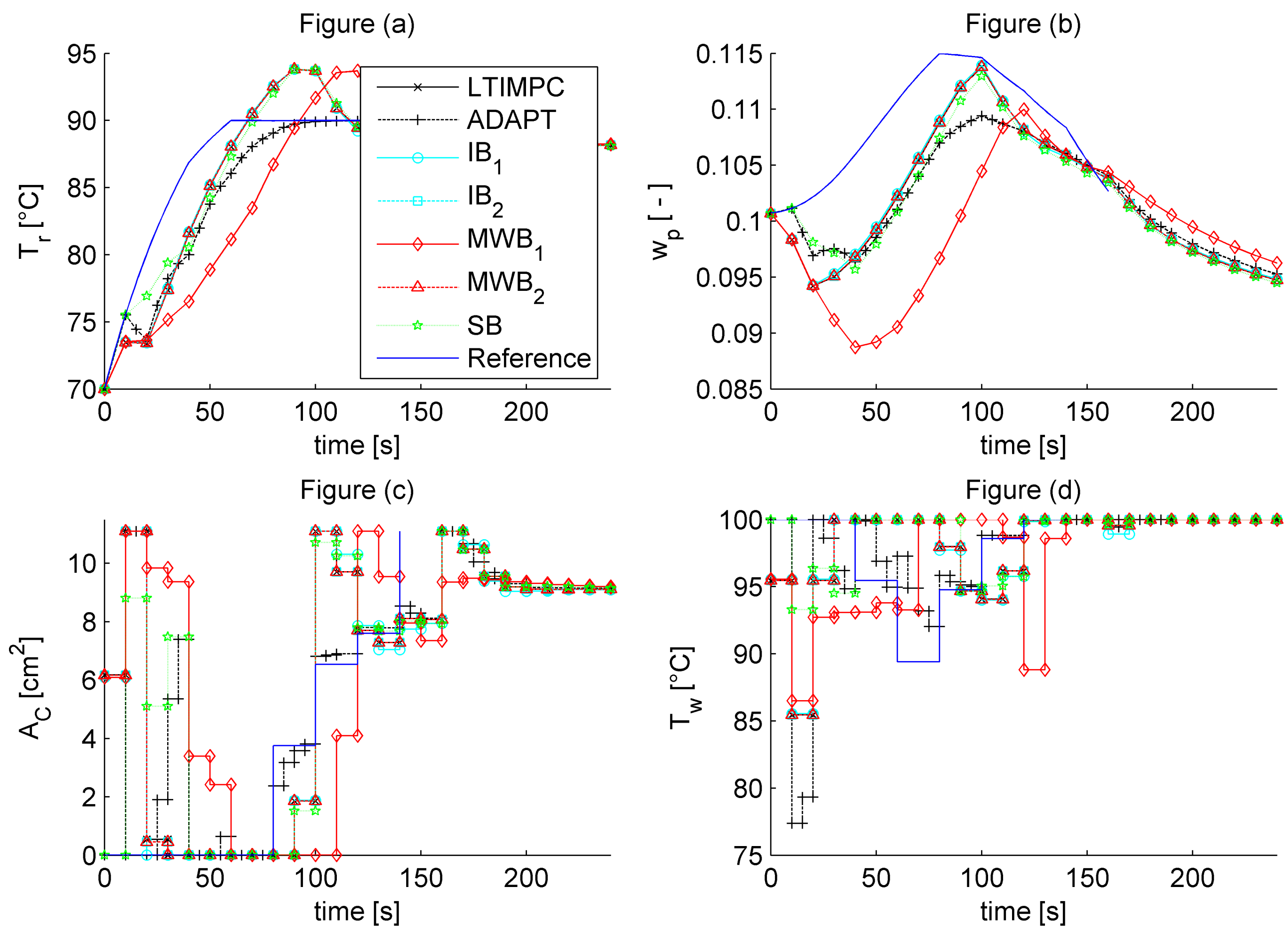

Figure 6.5: Results of all strategies for reference trajectory tracking performance scenario without disturbances. (a) Process output for the controlled variable, (b) Weight fraction w_p, (c) Control input A_c, (d) Control input T_w.

Table 6.6: Numerical results of the strategies for the reference tracking scenario without disturbances.

Abbreviation	Objective (%)	n_{DOF}	CPU time
LTIMPC	100	16+16	0.0092
ADAPT	92.00	6+6	n/a
IB_1	100.54	2+2	0.0036
MWB_1	135.71	2+2	0.0033
IB_2	100.81	10+9	0.0088
MWB_2	100.81	10+9	0.0086
SB	90.31	7+1	0.0047

6.3.2.2 Reference trajectory tracking with disturbances

In Table 6.5, the used strategies are summarized for the reference tracking performance analysis. The same strategies are used as for the set-point tracking scenario. However, the blocking map of *SB* is chosen differently because the dynamics differ from the set-point tracking scenario.

In Figure 6.6, the trajectories of the controlled output T_r (6.6a), the weight fraction w_p (6.6b), the control input A_c (6.6c) and the control input T_w (6.6d) are shown for every strategy listed in Table 6.5. In Table 6.7, the results for the simulations are given.

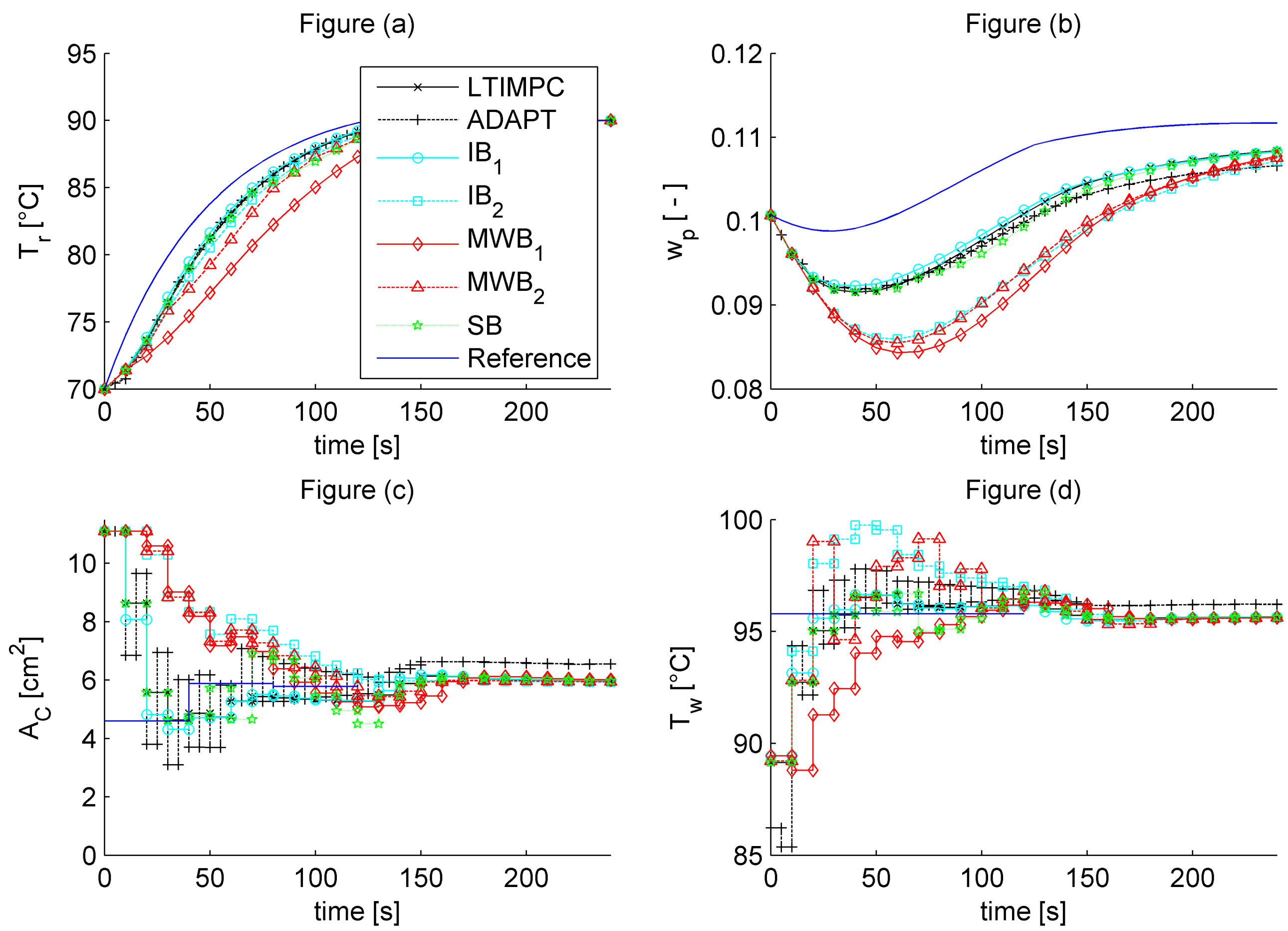

Figure 6.6: Results of all strategies for reference trajectory tracking performance scenario with disturbances. (a) Process output for the controlled variable, (b) Weight fraction w_p, (c) Control input A_c, (d) Control input T_w.

Table 6.7: Numerical results of the strategies for the reference tracking scenario with disturbances.

Abbreviation	Objective (%)	n_{DOF}	CPU time
LTIMPC	100	16+16	0.0108
ADAPT	102.06	5+4	n/a
IB_1	96.22	2+2	0.0029
MWB_1	143.13	2+2	0.0030
IB_2	106.11	10+9	0.0050
MWB_2	113.82	10+9	0.0038
SB	101.98	8+1	0.0036

6.3.2.3 Discussion of results for reference trajectory tracking

First, reference trajectory tracking without disturbances' results are illustrated in Figure 6.5 and numerical results are given in in Table 6.6. According to trajectory tracking without disturbances' results, *SB* has the smallest objective function value. *SB* reduces CPU time without heuristics and has better performance than MWB_2 which uses more n_{DOF} and $ADAPT$ which uses system dynamics. Besides, $ADAPT$ performs better than $LTIMPC$ as in Set-point tracking scenario.

Then, reference trajectory tracking with disturbances' results are illustrated in Figure 6.6 and numerical results are given in in Table 6.7. As expected the results in Table 6.7 and Figure 6.6 show that the CPU time reduces when reducing n_{DOF} as for the set-point tracking scenario. IB_1 has a better objective value than $LTIMPC$, although it has less n_{DOF}. This can be explained with the disturbances considered. When the closed-loop responses of these two strategies without disturbances are compared (from Table 6.6), IB_1 has almost same value with $LTIMPC$ and has not a better objective value than $LTIMPC$. As expected, MWB_1 shows the worst control performance as in the set-point scenario. The reason for this performance is the small number of degrees of freedom and the shift strategy because the first control input discretization is bigger than the next one.

In both scenarios (reference trajectory tracking and set-point tracking), *SB* has a better objective value than MWB_2 and $ADAPT$ even though they have shown a better control performance for the set-point tracking scenario. An important result is that *SB* reduces the CPU time without relying on heuristics and shows better control performance than MWB_2, *ADAPT* and IB_2 which use more n_{DOF}. This point shows that, when the system is not operated around a steady state, better results are obtained by using the dynamics of the system and shifting the blocking map of the system.

7. CONCLUSIONS AND RECOMMENDATIONS

For the real time implementation of MPC, the computational time to solve the optimal control problem is critical. All the calculations must be finished or terminated within the sampling time of MPC. In order to reduce the computational time, the number of decision variables for the control inputs can be reduced. To this end, adaptation and move blocking strategies can be applied.

In this thesis, an overview of the move blocking and adaptation techniques found in the literature are given. Move blocking is one of the methods for reducing the degrees of freedom resulting in the reduction of the computational burden. This reduction is done heuristically, i.e. control inputs are kept constant over several discretization intervals which usually coincide with the sampling time of the process with the help of a so-called blocking matrix. Generally, the generation of the blocking matrix or the implementation of blocking procedure is applied without knowledge of the system's dynamics. However, also blocking strategies which consider the system's behavior exist. Moreover, while the move blocking is applied for the inputs of the optimization problem, the inequality constraints can also be blocked to avoid feasibility problems. On the contrary, adaptation adapts the control structure, i.e. removes or adds grid points for the control input according to the dynamics of the system. Furthermore, for this purpose, adaptation uses mathematical tools such as wavelet analysis which adapts the trajectories and refines the grid by generally beginning from a coarse grid.

Move blocking strategies do not show any problems as regards on-line implementation, because the discretization-time intervals can be chosen in a way such that they are never smaller than the sampling time. Adaptation cab cause problems for on-line implementation because of the finest discretization of the control input trajectories which might be smaller than a sampling time. Then, the implementation of on-line dynamic optimization will be critical due to this finest discretization. Therefore, the state of the art of actuators were investigated. According to interviews with actuators producers or control valves producers, the technology of actuators and valves are capable of implementing fine control actions to the process. For adaptation, modulation actuators are important because old actuators restrict the refinement depths such that the discretization-time intervals must be a multiple of the sampling time. When new actuators are used, however, the optimization can continue until optimal value is reached. It can be concluded that they can realize an optimal control input trajectory for which the scan rate of the actuators must be higher than the sampling time of problem.

Finally, the performance analysis has shown that the computational time could be reduced by reducing the number of degrees of freedom with a little sacrificing the objective value. Furthermore, when the system's transient behavior has to be considered with reference trajectory tracking rather than set-point tracking, better results are obtained by using the dynamics of the system and shifting the blocking map of the system. Also, the suggested strategy *SB* shows good performance when comparing it to other strategies as regards the objective value and the computational time which is important point for this thesis. In future works, larger case studies which have more number of degrees of freedom and exhibit fast dynamics should be considered.

REFERENCES

Assassa, F., Brendel, M., Hartwich, A., Oldenburg, J., Schlegel, M., & Stockmann, K. (2013). DyOS 2.5 UsersGuide. Lehrstuhl für Prozesstechnik, Prof. Dr.-Ing. W. Marquardt, RWTH Aachen University, 2.5 ed.

AUMA (2013). Electric multi-turn actuators with integral controls. *AUMA Company*, Date retrieved 15.07.2013, address: http://www.auma.com/cms/Auma/en/products/controls/1,111003,19401.html

Betts, J., & Huffman, W. (1998). Mesh refinement in direct transcription methods for optimal control. *Optimal Control Applications & Methods*, Vol. **19**, pp. 1-21.

Binder, T., Blank, L., Bock, H., Bulirsch, R., Dahmen, W., Diehl, M., Kronseder, T., Marquardt, W., Schlöder, J., & von Stryk O. (2001). Introduction to Model Based Optimization of Chemical Processes on Moving Horizons. III. Springer Berlin Heidelberg.

Cagienard, R., Grieder, P., Kerrigan, E., & Morari, M. (2007). Move blocking strategies in receding horizon control. *Journal of Process Control*, Vol. **17**, pp. 563-570.

Canto, E., & Banga, J. (2000). Efficient optimal control of bioprocesses using second-order information. *Ind. Eng. Chem. Res.*, Vol. **39**, pp. 4287-4295.

Edgar, T. F., Smith, C., Shinskey, F., Gassman, G., Waite, A., McAvoy, T., & Seborg, D. (2008). Perry's Chemical Engineers' Handbook. McGraw-Hill, 8th ed.

Elixmann, D., Wolf, I., Schneider, R., Scheu, H., & Stuckert, M. (2013). DRTO Toolbox User's Guide. AVT-PT, RWTH Aachen.

Emerson, P. M. (2013). Module scan rate. Tech. rep., Date retrieved: 20.16.2013, address:http://socialkb.suite-sol.com/home/Emerson/DeltaV/Reduced _NEW_MasterBOL.ditamapExport-no-eps/Reduced_NEW_ MasterBOL.ditamap?nav=x151237file390-c _modulescanrate

Forbes, J. (1994). Model Structure and Adjustable Parameter Selection for Operations Optimization. Ph.D. thesis, McMaster University.

Freeman, H. (1965). Discrete-Time Systems; An Introduction to the Theory. John Wiley & Sons.

Gill, P.E., Murray, W., Saunders, M.A., & Wright, M.H. (1996). Snopt: an sqp algorithm for large-scale constrained optimization. Stanford University.

Gill, P.E., Murray, W., Saunders, M.A., & Wright, M.H. (1998). User s guide for npsol, version 5.0. systems optimization laboratory. Stanford University.

Gondhalekar, R., & Imura, J. (2010). Least-restrictive move-blocking model predictive control. *Automatica*, Vol. **46**, pp. 1234-1240.

Gondhalekar, R., Imura, J., & Kashima, K. (2009). Controlled invariant feasibility - a general approach to enforcing strong feasibility in mpc applied to move-blocking. *Automatica*, Vol. **45**, pp. 2869-2875.

Halldorsson, U., Fikar, M., & Unbehauen, H. (2005). Nonlinear predictive control with multirate optimisation step lengths. *IEE Proc.-Control Theory Appl.*, Vol. **152**, pp. 273-284.

Hangos, K., Bokor, J., & Szederkenyi, G. (2004). Analysis and Control of Nonlinear Process Systems. Springer.

Hannemann, R., Marquardt, W., Naumann, U., & Gendler, B. (2010). Discrete first and second-order adjoints and automatic differentiation for the sensitivity analysis of dynamic models. *International Conference on Computational Science*, Vol. **1**, pp. 297-305.

Hartwich, A. (2013). Adaptive Numerical Methods for Dynamic Real-time Optimization of Chemical Processes. Ph.D. thesis, Rheinisch-Westfalischen Technischen Hochschule Aachen.

Hartwich, A., Schlegel, M., Würth, L., & Marquard, W. (2007). Adaptive control vector parameterization for nonlinear model-predictive control. *International Journal of Robust and Nonlinear Control*, Vol. **00**, pp. 1-16.

Kvasnica, M., Grieder, P., & Baotic, M. (2004). Multi-parametric toolbox (MPT). Hybrid Systems: Computation and Control, *Lecture Notes in Computer Science*, Vol. **2993**, pp 448-462.

Lee, J., Chikkula, Y., Yu, Z., & Kontor, J. (1995). Improving computational efficiency of model predictive control algorithm using wavelet transformation. *Int. J. Control*, Vol. **61**, 859-883.

Li, S., & Petzold, L. (1999). Design of new daspk for sensitivitiy analysis. Technical report, Dept. of Mech. and Env. Eng. University of California.

MathWorks (2013). Quadratic Programming Algorithms. *The MathWorks, Inc.*, Date retrieved 21.06.2013, address: http://www.mathworks.de/de/help/optim/ug/quadratic-programming-algorithms.html

Mitsos, A. (2012). Applied numerical optimization lecture notes. RWTH Aachen.

Nocedal, J., & Wright, S. (1999). Numerical Optimization. Springer.

Olderwurtel, F., Gondhalekar, R., Jones, C., & Morari, M. (2009). Blocking parameterizations for improving the computational tractability of affine disturbance feedback mpc problems. *In Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*.

Oldewurtel, F. (2011). Stochastic Model Predictive Control for Energy Efficient Building Climate Control. Ph.D. thesis, Georgia Institute of Technology (dissertation ETH Zurich).

Palavajjhala, S., Motard, R., & Joseph, B. (1994). Blocking and condensing design for quadratic dynamic matrix control using wavelets. *Ind. Eng. Chem. Res.*, Vol. **33**, pp. 1159-1173.

Pannocchia, G., Rawlings, J., Mayne, D., & Marquard, W. (2010). On computing solutions to the continuous time constrained linear quadratic regulator. *IEEE Transactions on automatic control*, Vol. **55**, pp. 2192-2198.

Rawlings, J. B. (2000). Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, Vol. **20**, pp. 38-52.

Ricker, N. (1985). Use of quadratic programming for constrained internal model control. *Ind. Eng. them. Process Des. Dev.*,Vol. **24**, pp. 925-938.

Rosen, O., & Luus, R. (1992). Global optimization approach to nonlinear optimal control. *Journal of Optimization Theory and Applications*, Vol. **73**, pp. 547-562.

Schlegel, M. (2004). Adaptive discretization methods for the efficient solution of dynamic optimization problems. Ph.D. thesis, Rheinisch-Westfalischen Technischen Hochschule Aachen.

Schlegel, M., & Marquardt,W. (2006a). Adaptive switching structure detection for the solution of dynamic optimization problems. *Ind. Eng. Chem. Res.*, Vol. **45**, pp. 8083-8094.

Schlegel, M., & Marquardt, W. (2006b). Detection and exploitation of the control switching structure in the solution of dynamic optimization problems. *Journal of Process Control*, Vol. **16**, pp. 275-290.

Schlegel, M., Marquardt, W., Ehrig, R., & Nowak, U. (2004). Sensitivity analysis of linearly-implicit differential-algebraic systems by one-step extrapolation. *Applied Numerical Mathematics*, Vol. **48**, pp. 83-102.

Schlegel, M., Stockmann, K., Binder, T., & Marquardt, W. (2005). Dynamic optimization using adaptive control vector parameterization. *Computers and Chemical Engineering*, Vol. **29**, pp. 1731-1751.

Sclater, N. (2011). Mechanisms and Mechanical Devices Sourcebook. McGrawHill, 5th ed.

van Hessem, D. (2003). Implementation of the incoop control and optimization architecture. Delft University of Technology.

Van Loan, C. (2000). The ubiquitous kronecker product. *Journal of Computational and Applied Mathematics*, Vol. **123**, pp. 85-100.

Wachter, A., & Bieglery, L. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Subject Classification*, Vol. **106**, pp. 25-57.

Weissbach, T. (2007). Diskrete regelsysteme.

Williams, T., & Otto, R. (1960). A generalized chemical processing model for the investigation of computer control. *Chemical Processing Model for Computer Control*, November, pp. 458-473.

Wolf, I., & Marquardt, M. (2013). On adaptive move blocking for model-predictive control. Internal report. AVT-PT, RWTH Aachen.

APPENDICES

APPENDIX A

In this appendix, the transition matrices are given in the equation from **A.1** to **A.4** for construction of equations **2.36**.
With (for $n_{pred} = n_{con} = n$)

$$T_{[x_0 \to y]} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}, \tag{A.1}$$

$$T_{[u \to y]} = \begin{bmatrix} D & 0 & \cdots & \cdots & 0 \\ CB & D & 0 & \cdots & 0 \\ CAB & CB & D & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ CA^{n-2}B & \cdots & & & \end{bmatrix}, \tag{A.2}$$

$$T_{[d \to y]} = \begin{bmatrix} D_d & 0 & \cdots & \cdots & 0 \\ CB_d & D_d & 0 & \cdots & 0 \\ CAB_d & CB_d & D_d & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ CA^{n-2}B_d & \cdots & & & \end{bmatrix}, \tag{A.3}$$

$$T_{[y \to y]} = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 \\ CB_y & D & 0 & \cdots & 0 \\ CAB_y & CB_y & 0 & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ CA^{n-2}B_y & \cdots & & & \end{bmatrix}. \tag{A.4}$$

APPENDIX B

In order to linearize the state (eq.(**A.1**)) or the output (eq.(**A.2**)) equations, steady-state operation or reference points are needed to linearize around Hangos et al. (2004).

$$\begin{aligned} \dot{x} &= f(x,z,u,d,p,t), \qquad x(t_0) = x_0, & \textbf{(A.1)} \\ y &= g(x,z,u,d,p,t). & \textbf{(A.2)} \end{aligned}$$

In the DRTO toolbox the steady state operations points are defined as from equation (**A.3**) to (**A.6**).

$$\begin{aligned} \Delta x &= x - x_{ss}, \qquad \Delta x_0 = x_0 - x_{ss}, & \textbf{(A.3)} \\ \Delta u &= u - u_{ss}, & \textbf{(A.4)} \\ \Delta d &= d - d_{ss}, & \textbf{(A.5)} \\ \Delta y &= y - y_{ss} = y - g(x_{ss}, u_{ss}, d_{ss}). & \textbf{(A.6)} \end{aligned}$$

The principle of linearization is based on a Taylor series expansion of a (smooth) nonlinear function f around a steady state reference point. In the equation (**A.7**), the Taylor series expansion is shown. The linear approximation is the obtained by neglecting the higher order terms $\mathscr{O}^2$. In order to get the system matrices (A, B, C, D) of the linearized system equation **A.7** is used.

$$f(\dot{x}, z, x, u) = \underbrace{f(\dot{x}, z, x, u)|_{\mathrm{ss}}}_{=0} + J[\Delta\dot{x}^T, \Delta z^T, \Delta x^T, \Delta u^T]^T + \mathscr{O}^2 \qquad \textbf{(A.7)}$$

First term of the equation **A.7** equals to zero, because (e.g.) the equation $\Delta u = u - u_{ss}$ will be zero if we evaluate this function at u_{ss}, and this is also applied in the other variables.

The J represents the Jacobian matrix in the equation (**A.7**). Jacobian matrix (eq. (**A.8**)) is the function of f which contains the partial derivatives of the function with respect to the variables $\dot{x}$, x, u and d.

$$J = \left[\frac{\partial f}{\partial \dot{x}} \; \frac{\partial f}{\partial y} \; \frac{\partial f}{\partial x} \; \frac{\partial f}{\partial u}\right]_{\mathrm{ss}}. \qquad \textbf{(A.8)}$$

After neglecting $\mathscr{O}$ and using the equation **A.1** by equaling differential states to zero, the equation results in **A.9**.

$$0 = \left[\frac{\partial f}{\partial \dot{x}} \; \frac{\partial f}{\partial y} \; \frac{\partial f}{\partial x} \; \frac{\partial f}{\partial u}\right]_{\mathrm{ss}} [\Delta\dot{x}^T, \Delta z^T, \Delta x^T, \Delta u^T]^T \qquad \textbf{(A.9)}$$

As the OptoEcon-Toolbox considers only index-1 (if the differentiation of the algebraic equation is implicit ODE, this DAE systems call as index-1 DAE) problems, the matrix $J_{\dot{x},y} = \left[\frac{\partial f}{\partial \dot{x}} \; \frac{\partial f}{\partial y}\right]_{\mathrm{ss}}$ is regular and can be inverted. Hence, the linear equation **A.10** can be derived from **A.9**.

$$\begin{bmatrix} \Delta\dot{x} \\ \Delta z \end{bmatrix} = -\underbrace{\left[\frac{\partial f}{\partial \dot{x}} \; \frac{\partial f}{\partial y}\right]_{\mathrm{ss}}^{-1} \left[\frac{\partial f}{\partial x} \; \frac{\partial f}{\partial u}\right]_{\mathrm{ss}}}_{=\mathscr{S}=\begin{bmatrix} A & B \\ \bar{C} & \bar{D} \end{bmatrix}} \begin{bmatrix} \Delta x \\ \Delta u \end{bmatrix} \qquad \textbf{(A.10)}$$

The matrix $\mathscr{S}$ can be decomposed into the matrices A,B,$\bar{C}$,$\bar{D}$ as the size of the vector x is known. As the output variables y are only a subset of z, only the corresponding rows of ($\bar{C}$,$\bar{D}$) are required to build the matrices (C,D). And the system matrices will be in the equations **A.11** and **A.12**.

$$A^c = \left.\frac{\partial f}{\partial x}\right|_{(x_{ss},u_{ss},d_{ss})}, \quad B^c = \left.\frac{\partial f}{\partial u}\right|_{(x_{ss},u_{ss},d_{ss})}, \quad B_d^c = \left.\frac{\partial f}{\partial d}\right|_{(x_{ss},u_{ss},d_{ss})}, \tag{A.11}$$

$$C^c = \left.\frac{\partial g}{\partial x}\right|_{(x_{ss},u_{ss},d_{ss})}, \quad D^c = \left.\frac{\partial g}{\partial u}\right|_{(x_{ss},u_{ss},d_{ss})}, \quad D_d^c = \left.\frac{\partial g}{\partial d}\right|_{(x_{ss},u_{ss},d_{ss})}. \tag{A.12}$$

APPENDIX C

APPENDIX C.1 : Configuration of LTIMPC block

The number of control inputs n_{cont} as well as the number of control input predictions n_{pred} are 16 for the LTIMPC application without blocking. n_{cont} is changed according to move blocking strategy applied.

The weight matrices are constant in the regulatory objective function and determine how important the control inputs and controlled variables are. The weight matrices are denoted in (**5.1**) as Q and R. Q is the weighting matrix for the controlled variables and in this case study the controlled variable is the temperature of the reactor T_r. R is the weighting matrix for the control inputs which are cross sectional area of the valve A_c and the temperature of cooling water T_w. The weighting matrices Q and R are set to 1 and 0.1, respectively, for all move blocking strategies applied.

In the LTIMPC configuration file, the *simTimetoSS* represents the simulation duration in the gPROMS after with the system matrices are computed. This simulation duration should equal the maximal time when the system reaches the steady state. Here, the steady state time *simTimetoSS* is assumed to be 1200 s.

The *linearizeInEachSamplingTime* option is for integrating system from the current time to the end of simulation time. In the set-point tracking scenario (Section 6.3.1), the linearization is done only for the first cycle, around the set-point because the set-point values for the control inputs and controlled output are constant on all horizons. On the other hand, in the reference trajectory tracking scenario (Section 6.3.2), the linearization is done on every horizon because no set-point exists.

In the file $config_l timpc * .m$ ($*$ indicates the instance of the Ltimpc block):

```
% General MPC options
nPredictionSample = ...;    % int: number of predicted
 % output time samples
nControlSample    = ...;    % int: number of degrees
 % of freedom per input
                            % nControlSample <=
 % nPredictionSample
sampleTime        = ...;    % sample time of Linmpc
                            % normally the same as
 % for the process

simTimeToSs       = ...;    % simulation time to
 % steady state (until
 % this time assuming
 % to reach steady state)

% MPC weighting matrices
Q  = diag([...]);           % matrix (n_y x nR  =
```

```
 % diag([...]);
R = diag([...]);           % matrix (n_u x n_u):
 % weights for inputs
% MPC constraints
yLb  = [...];              % vector (n_y x 1):
 % lower bounds for outputs
yUb  = [...];              % vector (n_y x 1):
 % upper bounds for outputs
uLb  = [...];              % vector (n_u x 1):
 % lower bounds for inputs
uUb  = [...];              % vector (n_u x 1):
 % upper bounds for inputs
duLb = [...];              % vector (n_u x 1):
 % lower bounds for change
 % of inputs
duUb = [...];              % vector (n_u x 1):
 % upper bounds for change
 % of inputs
% model specifications: use "varGroupName",
%"varGroupVars", "varGroupType" to
% define custom variable groups for use
% within this specific block module:
varGroupName = 'CV';       % varGroupName='XYZ'
 % => there will be fields
                           % BlockModelInfo.nXYZ
 % and .nameXYZ
varGroupVars = {'...'};    % this should be
 % a list (or multiple lists) of
                           % variables belonging
 % to the custom groups.
varGroupType = 'Out';      % specify once for
 % each group. Can be 'Alg',
                           % 'Dif' or 'Dof'
 % or 'Inp' or 'Dis' or 'Out'
linearizeInEachSamplingTime = true;
 % if linearizeInEachSamplingTime
 % option included and set to
 % true, linearization will
 % be perofromed based on the
 % specified steady state
 % values in each time
 % step otherwise it will
   % be only performed in
 % the first iteration
```

APPENDIX D

APPENDIX D.1 : Configuration of Dynopt block

Besides tracking, the Dynopt block can also be used to find the economically optimal reference trajectories as discussed in Section 6.3. For this reason, two objective functions are defined in the model. The control objective is same as in the LTIMPC block with a regulatory control objective which tries to follow the reference or set-point trajectory. The economic objective is used for reaching a profit as high as possible.

Most of the properties are chosen to be the same for the regulatory and economic objective function. In the Dynopt block configuration file, the discretization time is set to 10 s and the on-line optimization sampling time is set to 10 s. In addition, the final time of the on-line dynamic optimization problem for every cycle is kept constant at 160 s as in the LTIMPC.

For DyOS, SNOPT is selected as the optimization program package. The configurations of the SNOPT can be found in the Appendix D. LIMEX is chosen as the integrator. Moreover, the starting numbers of grid refinement for adaptation is set to 4. Also, the adaptation settings are given in the both configuration files as grid refinement options.

The Dynopt block with the regulatory objective function minimizes the difference between the control inputs and the controlled outputs. The configuration file can be found in Appendix D. In addition, the Dynopt block with the economic objective function maximizes the profits by increasing the amount of the product P and E and decreasing the amount of the reactant A and B which are used in the reactor in (**6.27**). The configuration file can be found in Appendix D.

APPENDIX D.2 : SNOPT settings

SNOPT settings are given in Table D.1.

Table D.1: SNOPTS settings for DyOS.

Settings	
Begin ADOPT NLP problem	
Iterations Limit	10000
Major Optimality tolerance	1.0E-8
Verify Level	-1
Major Iterations Limit	5000
Minor Iterations Limit	5000
Major Feasibility Tolerance	1.E-6
Function Precision	1.E-8
Jacobian	sparse
Derivative Level	3
Linesearch tolerance	0.99
END	

APPENDIX D.3 : DyOS configuration files

APPENDIX D.3.1 : DyOS configuration file for regulatory objective

```
ESOPATH
wo wo

MINIMIZE
reactor.obj1

FINAL_TIME
160:0.0001:160

PIECEWISE_constant
reactor.FBIN_FL  # accually it is valve
# cross sectional area of valve cm^2
4 :  0.0001 : 11.1

reactor.Tw
4 : 20  : 100

INTERIOR_PATH_CONSTRAINTS
REACTOR.Tr
60 : 90

reactor.wp
0.105 : 0.2

#ENDPOINT_CONSTRAINTS
#reactor.ttime
#1000:1000

PARAMETER

reactor.FACTOR
0.9:0:0.9

REACTOR.TBIN
25:20:100

reactor.yconti
1 : 1 : 1   # says continuous stirred reactor

#reactor.tbin
#25:25:25

#reactor.tain
#25:25:25
```

```
OPTIONS
MODE : OPTIMIZATION
integrator:LIMEX
optimizer:SNOPT
GLOBALTOL : 6
OUTPUTLEVELSCREEN : 1
outputlevelfile: 1
#REMOVE_PREVIOUS_OUTPUT: OFF
PLOTGRID : 1
#PCRESOLUTION : 2

### Adaption
# grid refinement options:
ETRES   : 1.000000000e-2
# wavelet functions with smaller
# coefficients are removed
EPSILON : 0.9
# wavelets coefficients are marked
# and added up to epsilon for refinement
HORIZONTALREFINEMENTDEPTH : 1
VERTICALREFINEMENTDEPTH : 1
MAXSTEP : 10
EXITTHRESH : 1.000000000e-2
INTERPOLATIONSCALE : 10
MINSCALE : 2
MAXREFINEMENTSCALE : 10


### Inputfiles
finaltimeguess:finaltime.dat
##gridsource:initguesseco.dat
INITIALCONDITION : initdyos.dat
parametervalue:parameter.dat
#################################
```

APPENDIX D.3.2 : DyOS configuration file for economic objective

```
ESOPATH
wo_economic wo

MINIMIZE
reactor.obj

FINAL_TIME
500: 0.0001  : 500

PIECEWISE_constant
reactor.FBIN_FL
# accually it is valve cross sectional area A_c cm^2
```

```
1 :  0.0001  : 11.1

reactor.Tw
1 : 20  : 100

INTERIOR_PATH_CONSTRAINTS
REACTOR.Tr
60 : 90

reactor.wp
0.105 : 0.2

#ENDPOINT_CONSTRAINTS
reactor.ttime
500:500

PARAMETER

reactor.FACTOR
0.9:0:0.9

REACTOR.TBIN
25:20:100

reactor.yconti
1 : 1 : 1   # says continuous stirred reactor

OPTIONS
MODE : OPTIMIZATION
integrator:LIMEX
optimizer:SNOPT
GLOBALTOL : 6
OUTPUTLEVELSCREEN : 1
outputlevelfile: 1
#REMOVE_PREVIOUS_OUTPUT: OFF
PLOTGRID : 100
#PCRESOLUTION : 2

### Adaption
# grid refinement options:
ETRES   : 1.000000000e-2
# wavelet functions with smaller
# coefficients are removed
EPSILON : 0.9
# wavelets coefficients are marked
# and added up to epsilon for refinement
HORIZONTALREFINEMENTDEPTH : 1
VERTICALREFINEMENTDEPTH : 1
MAXSTEP : 10
```

```
EXITTHRESH : 1.000000000e-2
INTERPOLATIONSCALE : 10
MINSCALE : 2
MAXREFINEMENTSCALE : 10


### Inputfiles
#finaltimeguess:finaltime.dat
##gridsource:initguesseco.dat
#INITIALCONDITION : initdyos.dat
#parametervalue:parameter.dat
###################################
```

CURRICULUM VITAE

Name Surname: Canan DOMBAYCI

Place and Date of Birth: Balıkesir, 1988

Adress: İstanbul

E-Mail: dombaycicanan@hotmail.com

B.Sc.: Istanbul Technical Univesity, Chemical Engineering, (2006-2011)

M.Sc.: Istanbul Technical Univesity, Chemical Engineering, (2011-2013).