DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

FAST COMPUTATION STRATEGY FOR MODEL PREDICTIVE CONTROL

by

Barış DÜNDAR

August, 2020

İZMİR

FAST COMPUTATION STRATEGY FOR MODEL PREDICTIVE CONTROL

A Thesis Submitted to the

Graduate School of Natural and Applied Sciences of Dokuz Eylül University

In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in of Electrical and Electronics Engineering

by

Barış DÜNDAR

August, 2020

İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"FAST COMPUTATION STRATEGY FOR MODEL PREDICTIVE CONTROL"** completed by **BARIŞ DÜNDAR** under supervision of **ASSOC. PROF. DR. ÖZGE CİHANBEĞENDİ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

. . . .

Assoc. Prof. Dr. Özge CİHANBEĞENDİ

Supervisor

....

Assist. Prof. Dr. Özgür TAMER

Thesis Committee Member

....

Prof. Dr. Hasan ÖZTÜRK

Thesis Committee Member

....

Prof.Dr. Cüneyt GÜZELİŞ

Examining Committee Member

....

Doç.Dr. Savaş ŞAHİN

Examining Committee Member

Prof. Dr. Özgür ÖZÇELİK

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Assoc. Prof. Dr. Özge CİHANBEĞENDİ for his guidance through this thesis work, enlightening me with advices.

I would like to thank to my thesis examining committee members; Prof. Dr. Cüneyt GÜZELİŞ, Prof. Dr. Hasan ÖZTÜRK, Assist. Dr. Özgür TAMER, Assoc. Prof. Dr. Savaş ŞAHİN and Assist. Prof. Dr. Serkan GÜNEL who pointed out missing points and helped me to improve my thesis.

I thank to my wife Neslihan and my family for their support and patience.

I also want to thank to academic and administrative staff of the Graduate School of Natural and Applied Sciences of Dokuz Eylül University for their valuable support.

Barış DÜNDAR

FAST COMPUTATION STRATEGY FOR MODEL PREDICTIVE CONTROL

ABSTRACT

This thesis focuses on slow operation issues of the existing Model Predictive Control (MPC) algorithms and proposes a method to speed the slow response issues (rise time, settling time and prediction) of them up due to the computation load of online optimization without reducing the computation load.

In the proposed method of this thesis, the original MPC strategy is warped using an integer time scaling factor in time domain. The aim is to speed this strategy up by exploiting the speeds of fast microprocessors. For this aim an integer time scaling factor is selected according to the limit of the used microprocessor. The solution of the plant is contracted and sampled by this scaling factor. Standard MPC problem is based on this modification of the plant. First elements of fast computed control law vectors are dilated until original online optimization sampling period is reached again and applied to plant at each sampling period.

Modified MPC strategy, which is called *Upsampled MPC (UMPC)*, settles faster to desired output than original MPC strategy and also satisfies faster prediction at each sampling period. Since the method is more compatible with fast microprocessor and speed of the UMPC results from microprocessor, slow operation issue of MPC algorithms is solved for different time scaling factors by this method. Therefore, control energy of UMPC is lower. There also exists free times between online optimization sampling periods of original MPC and UMPC strategies. These free times can be used for improvement of existing MPC algorithms.

Keywords: Model predictive control, linear and nonlinear dynamical systems, time scaling plant model, fast settling response, free time period, low control energy

MODEL ÖNGÖRÜLÜ KONTROL İÇİN HIZLI HESAPLAMA STRATEJİSİ

ÖZ

Bu tez, var olan Model Öngörülü Kontrol (Model Predictive Control (MPC)) algoritmalarının yavaş çalışma sorunlarına odaklanmakta ve bu algoritmaların çevrimiçi optimizasyon hesaplama yükü sebebiyle yavaş cevap sorunlarının (yükselme zamanı, yerleşim zamanı ve tahmini) hesaplama yükünü azaltmadan hızlandırmak için bir yöntem önermektedir.

Bu tezin önerilen yönteminde, orijinal MPC stratejisi, zaman domeninde bir tamsayı olan zaman ölçeklendirme faktörü kullanılarak bükülür. Amaç, hızlı mikroişlemcilerin hızlarından yararlanarak bu stratejiyi hızlandırmaktır. Bu amaç için kullanılan tamsayı zaman ölçeklendirme faktörü kullanılan mikroişlemcinin sınırına göre seçilir. Sistemin çözümü bu ölçekleme ile büzülür ve örneklenir. Standart MPC problemi, sistemin bu modifikasyonuna dayanmaktadır. Hızlı hesaplanan kontrol kanunu vektörlerinin ilk bileşenleri orijinal çevrimiçi optimizasyon örnekleme periyoduna kadar genişletilir ve bu periyotlarda sisteme uygulanır.

Upsampled MPC (UMPC) olarak adlandırılan modifiye edilmiş MPC stratejisi orijinal MPC stratejisine göre istenilen çıkış değerlerine daha hızlı yerleşir ve aynı zamanda her bir örnekleme periyodunda daha hızlı bir tahmin sağlamaktır. Yöntem hızlı mikroişlemci ile daha uyumlu olduğundan ve UMPC'nin hızının kaynağı mikroişlemci olduğundan, bu metodla farklı zaman ölçekleme faktörleri için MPC algoritmalarının yavaş çalışma sorunu çözülür. Bu nedenle, UMPC'nin kontrol enerjisi daha düşüktür. Ayrıca, orijinal MPC ve UMPC stratejilerinin çevrimiçi optimizasyon örnekleme periyodları arasında serbest zaman periyodları oluşmaktadır. Bu serbest zaman periyodları, mevcut MPC algoritmalarının iyileştirilmesi için kullanılabilir.

Anahtar Kelimeler: Model öngörülü kontrol, lineer ve lineer olmayan dinamik sistemler, zaman ölçeklemeli sistem modeli, hızlı oturma cevabı, serbest zaman periyodu, düşük kontrol enerjisi

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER 1

INTRODUCTION

In control of dynamical systems, optimal control theory is a dynamical mathematical optimization (or mathematical programming) method, which is maximization or minimization of an scalar objective, cost function or performance index by determining optimal input function under their physical constraints (input, input rate, state and output) and its basics were presented by Lev Semyonovich Pontryagin and Richard Bellman in 1950s.

In optimal control theory, unconstrained infinite horizon Stochastic Linear Quadratic Regulator (LQR) or Linear Quadratic Gaussian (LQG) optimal control problem, which guarantees stability, was published by Rudolf E. Kalman in 1960s and it may be assumed as the first step for the development of modern control concepts and the fundamental of Model Predictive Control (MPC) algorithm.

The optimal control methods are offline and precomputed optimal control methods. Since these methods have the fixed control laws for all states, they cannot adapt to changing situations due to disturbances, but MPC is an online optimal control method over receding finite time horizon. It performs online optimization using current and future predictions based on well - defined mathematical model of plant, which will be defined plant model. Since MPC operates online, it is superior than offline methods and successful on physical constraints of plant, which are handled in the MPC optimization computations. Another important fundamental concept for MPC algorithm is Receding Horizon Principle (RHP) or Receding Horizon Control. For RHP Propoi presented the theoretical work in 1963. This strategy compensates modelling errors and disturbances. The first description of MPC algorithm was expressed in Model Predictive Heuristic Control (MPHC) approach by Richalet at 1976. Several years later Cutler and Ramaker presented Dynamical Matrix Control (DMC). DMC was very successful in the petro-chemical industry which operates slowly (Qin & Badgwell, 2003; Moor, 2017; Taha, 2017, 2015; Johansson, 2015; Zak, 2013b; Kirk, 2004; Ratkovic, 2016; Tits, 2018).

1.1 General Optimal Control Methods and Model Predictive Control

An optimal control problem can be solved by using Dynamical Programming (DP), Indirect and Direct Methods (IM and DM), which have some limitations, offline (alternatively explicit or precomputed) and also online (alternatively real time or implicit) in Model Predictive Control strategy (Zak, 2013b; Kirk, 2004; Ratkovic, 2016).

DP is formulated by continuous or discrete - time partial Hamilton-Jacobi-Bellman differential equations (Kirk, 2004). In DP, a high - dimensional complex optimal control problem is divided into less complex sub and small - dimensional optimal control problems and these problems are solved by working backwards from starting terminal state at each iteration. For example the optimization of a previous sub - cost function from the terminal cost is represented by combining current and terminal costs and optimized by checking second - order sufficient condition, as well. All the optimized sub problems are combined. In this computation procedure, computing the same sub - problems is avoided and continued until reaching the solution of the original optimal control problem.

IMs are based on Pontrayegin's Minimum Principle (PMP). In the method, the calculus of variations is used to compute the first-order necessary optimality conditions (Euler-Lagrange Equations) of the optimal control problem, which converts to a two-bound (or multiple-point) boundary value problem and thus the original optimal control problem is solved indirectly. Hamiltonian system in this problem is solved and local minimum, maximumum or saddle control laws obtained. Hamiltonian function is minimized in continuous time. Optimal control problem in IMs is first optimised and then discretizated using shooting, multiple shooting or discretization.

In DMs, state and input of optimal control problem is directly discretized and the discrete-time optimal control problem is rewritten as Nonlinear Programming (NLP) problem. Infinite optimal control problem in continuous time is converted to finite optimal control problem in discrete time and this provides less computation effort. NLP is solved by numerical methods faster and effective. Lagrangian function is minimized in discrete time.

Before going into the details of the MPC algorithm, the advantages and disadvantages of the optimal control methods are briefly considered:

- Computational complexity of DP for high - dimensional systems is high and more global approach than PMP but to finding optimal control law for the same conditions is generally very difficult and to applying both approaches to plants with small states are more convenient,
- DP satisfies necessary and sufficient conditions without requiring linearity / convexity and its solution is a global minimum but PMP requires and solution of PMP is a global minimum under linearity / convexity,
- Since DP and PMP have the precomputed control laws for all possible states on plant model and, in practice, the control law required for controlling plant can be changed in the presence of disturbances and mismatch between plant and plant model, to apply them to plant with maximum 10 states are impossible,
- DP has a simpler computation logic than the mathematical computation difficulty of PMP,
- Depending on the nature of optimal control problem, DP or PMP can be preferred,
- In continuous time, the optimal control law of PMP can have a discontinuous function depending on state and co - state variables. That can cause non - smooth differential boundary value problem, which results high - level non - linearity and instability. Therefore, this approach can be characterised better in continuous time (Diehl & Gros, 2017),
- The optimality conditions of Direct and indirect methods are related to each other and uses discretisation (Ratkovic, 2016). To implement Newton methods on the interior point and active set algorithms for linear dynamical systems and these algorithms for quadratic (convex) optimal control problems are quite effective and faster,
- Direct Methods are faster and effective than both methods,
- Since Model Predictive Control with direct method computes repetitive or updated control laws in receding horizon fashion and satisfies prediction of future using plant

model on constraints online at each sampling period, it is the most effective control method for high - scale MIMO dynamical systems in optimal control theory (Rao, 2015; Ha, 2017; Diehl, 2015; Katja & Felis, 2011; Bertsekas, 2005; Stengel, 2010; Diehl & Gros, 2017; Diehl et al., 2009; Zeilinger & Jones, 2011),

- DP and PMP for small - dimensional dynamical systems can be alternatively implemented offline or online in Model Predictive Control (Pena et al., 2004; Xie et al., 2018).

In industry and academic environments, the research topics in the design and implementation of MPC are focused to improve the following main frameworks

- design formulations for robust closed - loop stability and feasibility,
- handling methods hard and/or soft physical constraints of MIMO dynamical systems, which cause nonlinearty and deteriorate the trajectories of the computed control laws and also slow down to find the optimal control,
- the performance of online optimization,
- fast, low-cost, reliable, robust, simplicity and efficiency of the optimal control system.

The computation principle for all the formulations of MPC algorithms is the same and the design procedure of an MPC system consists of five important items (Wang, 2008):

- Plant model and disturbance model,
- Objective or cost function,
- Constraints,
- Optimization,
- RHP.

In the literature, the MPC algorithms are substantially successful on linear dynamical systems and all the researches focus on the fast-optimized closed-loop stability of linear and nonlinear dynamical systems taking advantages of other control techniques.

The closed - stability and recursive feasibility properties of MPC algorithms for linear time-invariant (LTI) dynamical systems were satisfied by tuning horizon, weighted matrices of quadratic state and input costs in 1980s as case by case but this is expensive and unreliable for all time. There does not exist closed-loop control laws over receding horizon fashion. In 1990s, its formulation was modified by using stability properties of LQR problem, which guarantees asymptotically stability of unstable plant for all time. In this method, a terminal cost and terminal constraints are included in the MPC problem. The weighted matrix of the terminal cost is computed using Lyapunov equation and the feedback gain computed using LQR problem. After determining the matrix, recursively feasibility is checked.

The approaches are described as Nominal (Standard or Classical) MPC algorithm and the closed-loop stability and feasibility of this algorithm is substantially and theoretically guaranteed. In the literature of MPC, "Robust MPC" and "MPC for Tracking" variations are established under the fundamental idea of Nominal MPC but theoretical establishment of MPC algorithms for stochastic and nonlinear systems are still a research motivation.

MPC has evolved to dominate the process industry and we evaluate that it is superior than other control strategies (Kouvaritakis et al., 2016; Garcia et al., 1989; Boom et al., 2007)

In practical or real-time applications, inspite of the fact that stability properties of MPC algorithms are theoretically well established, its implementation on hardware is not successful or limited for controlling dynamical systems with fast - sampled period. This is a research topic. After an MPC problem is converted to a Nonconvex Nonlinear Programming, which is Quadratic Programming (QP) for linear systems, it is solved using the improved numerical methods. Current and future inputs (nonlinear control laws) and output responses of plant are predicted at each sampling period. When the existing MPC algorithms contain problems with high-dimensional plant models and constraints, long state and control horizons, and high sampling times, control of plant becomes to be difficult (Kouvaritakis et al., 2016). Thus, the MPC algorithms are convenient to be used for slow dynamics because of the computational load. Computational load causes slow response (slow rise time, settling time and prediction), feasibility, stability and timing problems in controlling linear and nonlinear plants with high - sampled period. In this case, the theoretical MPC problems for feasibility and closed - loop stability are not applicable in real - time. The most important reasons for the slow response are as follows:

- Inclusion of hard and soft physical constraints,
- QP and NLP solvers can not converge to an optimal and feasible solution, to a feasible local or global minimum solution and the required time to find a minimum solution can take a long time,
- Non-sparse high-dimensional matrix multiplications and growing computation time with state and control horizons,
- High-dimensional prediction model and physical constrains,
- Control requirements with short sampling periods,
- Time elapsed on online system identification, if it is available in control (Borrelli et al., 2015; TøNdel et al., 2003; Rao et al., 1998; Bemporad et al., 2002; Cai et al., 2014; Borrelli et al., 2010; Wang & Boyd, 2010; Yildirim & Wright, 2002; Koehler et al., 2017; Morari, 2013; Zeilinger, 2011; Liu et al., 2018; Ling & Maciejowski, 2009; Kirk, 2004).

In the literature, with the improvements of fast and parallel processors in the 1990's, the online computational load can be partially reduced by the following remarkable explicit and implicit MPC methods;

- Considering the prediction plant model states as the online optimization variable in QP (Kouvaritakis et al., 2016),
- Using the microprocessors or FPGA ICs, which operate in parallel and faster (Jerez et al., 2011; Ling et al., 2006, 2008),

- Using the improved warm - start implicit and explicit MPC methods (active set, interior point, and multi-parametric solvers) (Borrelli et al., 2015; TøNdel et al., 2003; Rao et al., 1998; Bemporad et al., 2002; Fletcher, 2013; Cai et al., 2014; Borrelli et al., 2010; Wang & Boyd, 2010; Yildirim & Wright, 2002; Koehler et al., 2017; Morari, 2013; Zeilinger, 2011; Liu et al., 2018; Ling & Maciejowski, 2009),

- Reducing dimensions of matrices and vectors and to simplifying multiplication using the sparsity formulations of optimization problems, if it is possible (Qi et al., 2015),

- Shrinking state and control horizons in the online optimization time, if it is possible (Dunlap et al., 2008),

- Implementing the combination of the slow and fast MPC algorithms to plant, which can be divided into slow and fast subsystems (model reduction techniques) (Zhang et al., 2013).

1.2 Upsampled Model Predictive Control for Linear and Nonlinear Dynamical Systems and Contribution

In this thesis, we propose a method to speed the slow response issues (rise time, settling time and prediction) of the existing MPC algorithms up due to the computation load of online optimization without reducing computation load. In the method, original MPC strategy, which is nominal and basic MPC algorithm, is warped using an integer time scaling factor in time domain. The aim is to speed this strategy up by exploiting the speeds of fast microprocessors. For this aim an integer time scaling factor is selected according to the limit of the used microprocessor.

The solutions of linear and feedback linearized dynamical systems, which are controlled and will be called *plant*, is contracted and sampled by this scaling factor. The nominal MPC problem is constructed on this modification of plant, which will be called *upsampled plant model (UPM)*. The first elements of fast computed control law vectors in the method are dilated by storing their values until the original online optimization sampling period is reached again and applied to plant at each sampling period. The modified MPC strategy settles faster to desired output than original MPC strategy and also satisfies faster prediction at each sampling period. Since the method is more compatible with fast microprocessor and the speed of UMPC results from microprocessor, slow operation issue of MPC algorithms for controlling high - speed dynamical systems is solved for different time scaling factors by this method. Therefore, the control energy of UMPC is lower. There also exists the free times between online optimization sampling periods of original MPC and UMPC strategies.

Since the modified MPC is the compressed version of the original MPC strategy in the time domain, we will call *Upsampled MPC (UMPC)*. Since UPM has faster sampling period than the original plant model, the term "upsampled" is used. This expression is not related to the terms "upsampling" used signal processing, which is the process of adding zero - valued samples between original samples to increase the sampling rate.

As the another contribution, the free times can be used;

- to compensate modelling errors and disturbances by online system identifications,
- to incorporate with adaptive control and other control strategies,
- to increase control capabilities by different scenarios.

When the free times are not used for the improvement of MPC algorithms, UMPC has energy efficiency according to the original MPC. The method can be applied to the MPC methods (Jerez et al., 2011; Ling et al., 2006, 2008; Borrelli et al., 2015; TøNdel et al., 2003; Rao et al., 1998; Bemporad et al., 2002; Fletcher, 2013; Cai et al., 2014; Borrelli et al., 2010; Wang & Boyd, 2010; Yildirim & Wright, 2002; Koehler et al., 2017; Morari, 2013; Zeilinger, 2011; Liu et al., 2018; Qi et al., 2015; Dunlap et al., 2008; Zhang et al., 2013) and nonlinear dynamical systems.

1.3 Outline of the Thesis

In Chapter 1, the general optimal control techniques are briefly summarized. The similarities and differences of these techniques to Model Predictive Control and the advantages and disadvantages between them are compared. In this thesis, our main focus is that MPC can be applied to fast dynamics. In the literature, some fast MPC methods has been described to speed online slow computation burden up. However, these have significant limitations. In Chapter 1, the current issues of the optimal control and MPC methods are discussed and explained how our approach can solve them.

Chapter 2 contains the basic mathematical concepts, definitions and theorems needed, as well as information on the solution of control theory and nonlinear optimization problems or nonlinear programming (NLP). NLP problems are divided into two category. These are unconstrained and constrained NLP problems. In Chapter 2 the solution methods for the problems are explained.

In Chapter 3, we propose our method and define the time - scaled and upsampled forms of LTI systems, which are based on a integer time scaling factor selected according to the speed of the microprocessor used and then introduce the UMPC strategy based on the upsampled plant model, which is compressed in time domain. In the method, it is described how the contracted and upsampled plant models are obtained and then UMPC problem is formulated. The performances of the original MPC and UMPC are illustrated on the example and the results are compared.

In Chapter 4, the linearization method of the nonlinear systems, which are commonly used in the practical control applications, is combined with the UMPC approach. The UMPC problem based on the linearized upsampled plant model is constructed. Similarly, the performances of the original MPC and UMPC are illustrated on the example and the results are compared.

Finally, in Chapter 5, the main conclusions of the UMPC are emphasized and the information about future studies is given.

CHAPTER 2

BACKGROUND

This chapter presents basic and general mathematical tools, necessary definitions and theorems, which are available in the literature, utilized in this thesis and we do not give an analysis. Refer to the references (Kall, 2015; Tibshiran, 2017; Tibshirani, 2017; Gordon & Tibshirani, 2012; Burke, 2020; Boyd & Vandenberghe, 2009; Hein, 2010; Hedengren, 2020; Robert, 2013; Ghaoui, 2012; Lars, 2018a,b; Singh & Ravikumar, 2017; Forsgren et al., 2002; Boyd & Vandenberghe, 2004; Bubnicki, 2005; Chen, 1999; Sastry, 1999; Ling & Maciejowski, 2009) for the details.

2.1 Mathematical Concepts

2.1.1 Set Theory

Definition 1 (Convex Set)**.** *A set $\mathcal{X} \subseteq \mathbb{R}^n$ is convex if $\alpha x_1 + (1-\alpha)x_2 \in \mathcal{X}$ for any $x_1, x_2 \in \mathcal{X}$ and $\alpha \in [0, 1]$*

Definition 2 (Affine Set)**.** *A set $\mathcal{X} \subseteq \mathbb{R}^n$ is affine if $\alpha x_1 + (1-\alpha)x_2 \in \mathcal{X}$ for any $x_1, x_2 \in \mathcal{X}$ and $\alpha \in \mathbb{R}$*

Definition 3 (Normed Linear Spaces)**.** *The norms of a set $x \in \mathcal{X} = \mathbb{R}^n$ are given by*

- *The sum or 1 - norm, ℓ_1, $\| \ x \ \| \triangleq \sum_{i=1}^{n} \mid x_i \mid$ if $(\mathcal{X}, \| \ . \ \|)_1$,*
- *The euclidian norm, ℓ_2, $\| \ x \ \|_2 \triangleq \sqrt{\sum_{i=1}^{n} \mid x_i^2 \mid} = \sqrt{x^T x}$ if $(\mathcal{X}, \| \ . \ \|)_2$,*
- *The p - norm $\| \ x \ \|_p \triangleq (\sum_{i=1}^{n} \mid x_i^{1/p} \mid)^{1/p}$ if $(\mathcal{X}, \| \ . \ \|)_p$ where $p \geq 1$,*
- *The ∞ - norm $\| \ x \ \|_\infty \triangleq \max_i \mid x_i \mid$ if $(\mathcal{X}, \| \ . \ \|)_\infty$*

where the norm space $\| \ . \ \|_n$: $\mathcal{X} \mapsto \mathbb{R}_+$ for $n = 1, 2, ...$

Definition 4 (Weighted $2-$ Norm)**.** *The weighted matrix norm $\| \ x \ \|_A = \sqrt{x^T A x} = \| A^{1/2} \ \|_2$ where $A \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix if $(\mathcal{X}, \| \ . \ \|)_A$.*

Definition 5 (Ball)**.** *The open ball ϵ in $\mathbb{R}^n$ around a given point $x_a \in \mathbb{R}^n$ is the set*

$\mathcal{B}_\epsilon \triangleq \{x \in \mathbb{R}^n \mid \| x - x_a \| < \epsilon\}$*, where the radius $\epsilon > 0$ and $\| \, . \, \|$ denotes any vector norm, usually the Euclidian norm $\| \, . \, \|_2$*

Definition 6 (Smooth Maps)**.** *A map $f : \mathcal{X} \mapsto \mathcal{V}$ is called smooth if all of the partial derivatives $\partial f / x_{i_1}, ..., x_{i_n}$ exist and are continuous.*

Definition 7 (Diffeomorphism)**.** *A map $f : \mathcal{X} \mapsto \mathcal{V}$ is called a diffeomorphism iff f is a homeomorphism (i.e., a one - to - one continuous map with continuous inverse) and if both f and f^{-1} are smooth..*

2.1.2 Polyhedra

Definition 8 (Halfspace)**.** *A closed halfspace in $\mathbb{R}^n$ is a set of the form*

$\mathcal{H}_\epsilon = \{x \in \mathbb{R}^n \mid a^T x \geq b\}$, $a \in \mathbb{R}^n$, $b \in \mathbb{R}$

Definition 9 (Polyhedron)**.** *A polyhedron is in $\mathbb{R}^n$ is the intersection of a finite number of halfspaces.*

2.1.3 Function Theory

Definition 10 (Continuous Function)**.** *A function $f : \mathcal{D} \mapsto \mathbb{R}^n$ with the domain $\mathcal{D} \subseteq \mathbb{R}^n$ is called continuous at a point $x^* \in \mathcal{D}$ if*

$$\|x - x^*\| < \delta(\epsilon) \;\Rightarrow\; \|f(x) - f(x^*)\| < \epsilon \;\; \forall \epsilon,$$

f is called continuous.

Definition 11 (Lipschitz Continuity)**.** *A function $f : \mathcal{D} \mapsto \mathbb{R}^n$ with the domain $\mathcal{D} \subseteq \mathbb{R}^n$ is called Lipschitz continuous if*

$$\|f(x) - f(x^*)\| \leq K\|x - x^*\| \;\; \forall x, x^* \in \mathcal{D},$$

for some $K \in \mathbb{R}$ is called a Lipschitz constant.

Definition 12 (Polyhedral Piecewise Affine (PWA) Function). *A function $f : \mathscr{P} \mapsto \mathbb{R}^n$ with $\mathscr{P} \subseteq \mathbb{R}^n$ is piecewise affine (PWA) if $\mathscr{P}^n$ is a polyhedral partition of $\mathscr{P}$ and $f(x) \triangleq C^i x + D^i$ if $x \in \mathscr{P}_i \forall \mathscr{P}_i \in \mathscr{P}^n$, where $C^i \in \mathbb{R}^{n \times n}$, $D^i \in \mathbb{R}^n$ and $j \in 1, \ldots, N$.*

Definition 13 (Convex and Concave Function). *A function $f : \mathscr{D} \mapsto \mathbb{R}^n$ is convex, if its domain $\mathscr{D} \subseteq \mathbb{R}^n$ is a convex set and*

$$f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2) \tag{2.1}$$

for any $x_1, x_2 \in \mathscr{D}$ and $\alpha \in [0, 1]$. $f(.)$ is concave if -$f(.)$ is convex. A function f is strictly convex if strict inequality holds in the inequality 2.1 whenever $x_1 \neq x_2$ and $0 < \alpha < 1$.

2.2 Control Theory

The control problems has been characterized by four important types of process variables. These are controlled variables (CVs) or output variables "y", manipulated variables (MVs), input, control input or excitation variables "u", state variables (SVs) "x" and disturbance variables (DVs) "d" which are the functions of time. In modern control theory, state variables are used. State variables are the internal variables of plant or dynamical systems that are represented by first order state differential equations or state equations. In the process control, output variables are desired to set a reference variable or set point "r" and a control input for that goal is computed. r and d are external inputs and u and y are internal signals. Input variable for the objective of control must be computed. Computed input variables are called *control law* or *controller*. Generally, control law for the deterministic systems is a function of state variable given by linear time invariant form $u(t) = Fx(t)$ with constant matrix F and nonlinear time invariant form $u(t) = \mu(x(t))$, and the type of control is called *closed loop* or *feedback control*. The closed - loop systems are the negative feed - back systems, which has a stabilizing character (Bubnicki, 2005). The control law must satisfy stabilization of a dynamical system or process at operating conditions.

In the literature and industry, there are two main structures of control systems embedded in a microprocessor (μP) for process control. Figures 2.1a and 2.1b shows model - based and classical control structures or control systems, respectively. M is the mathematical model of process or plant P. For any pair of external inputs $\{r, d\}$ and the same internal signals $\{u, y\}$, they are equivalent but Figure 2.1a, which is referred as Internal Model Control (IMC) (Garcia et al., 1989), is more attractive than Figure 2.1b. Under perfect model $P = M$, the process may be controlled on the model M embedded in a processor. In modern control theory, the both may be represented with state variable vectors as shown in Figure figure 2.1c and Figure figure 2.1d shows general control perspective.

The controller C is a set of controllers. *Proportional – Integral – Differential (PID), Linear Quadratic Regulator (LQR)* and *Model Predictive Control (MPC)* for C are the most widely used the basic control techniques in process control. The control techniques are improved by adaptive control techniques, stochastic and robust analyses. In control theory, PID is a classical control technique because it operates by three setting parameters (proportional, integral and derivative) and is simple but generally, it may not success for MIMO dynamical systems. PID controllers are practical, simple, offline and non - optimal controllers for especially unconstrained and small - dimensional linear dynamical systems. In addition, there does not exist optimal energy computation and prediction capability for the future behaviours of the process. There is directly control computation dependence on the process at current time because using it is not feasible controlling critical systems (such as aerospace and chemical industries). The disadvantages of PID controllers are compensated by LQR, which is an optimal control method but it is not effective on constraints of the process. This controller is unconstrained and infinite time horizon optimal control for linear dynamical systems.

In LQR problem, the control law is computed offline and is the same for all states and times. If we use process model and handle the physical constraints of process in finite time horizon, it is called *the finite horizon constrained optimal control problem* and if the optimal control problem for high – scale processes is solved at each sample time and on receding time horizon, the optimal control problem is called MPC, which uses the structure in Figure 2.1a. In RHP strategy, the control laws vectors are computed by the optimization based on the prediction plant model and updated at each

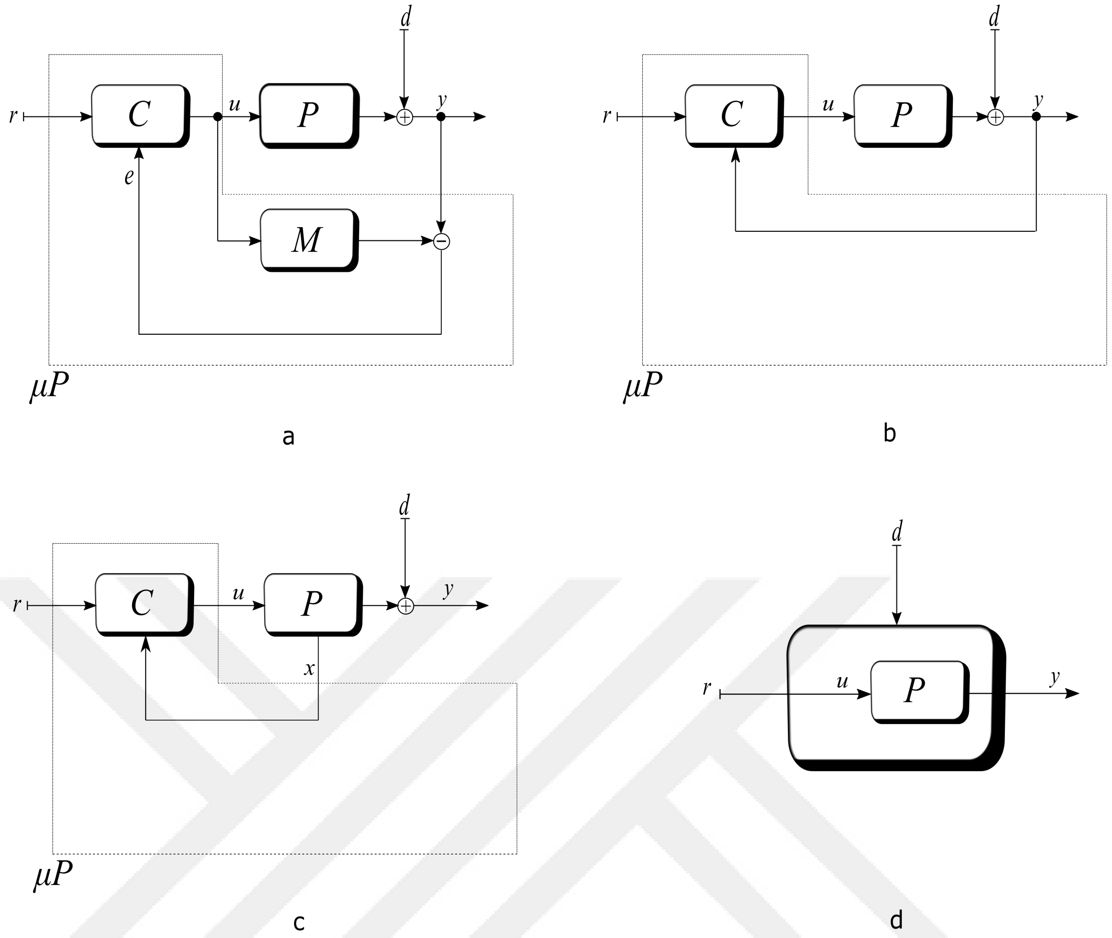

Figure 2.1 The Structures of control systems (1a) model - based control structure (1b) classical control structure (1c) control system representation with state variable (1d) General control perspective (Garcia et al., 1989)

control sampling period over receding (moving) time horizon. The first components of computed control law vectors are applied to the plant at each current time instant. The more explicit form of MPC block structure and receding horizon time strategies are given in Figure 2.2 and 2.3. Since the process P is a physical dynamical system, we assume that it has unique solution and is controllable by $u = u(t)$ for its stabilization. In the opposite case, we can not discuss controlling a process.

Under this assumption, the process P may have the following properties:

- Deterministic linear or nonlinear,
- Stochastic linear or nonlinear,
- Single - Input and Single - Output (SISO) or Multi - Input and Multi - Output

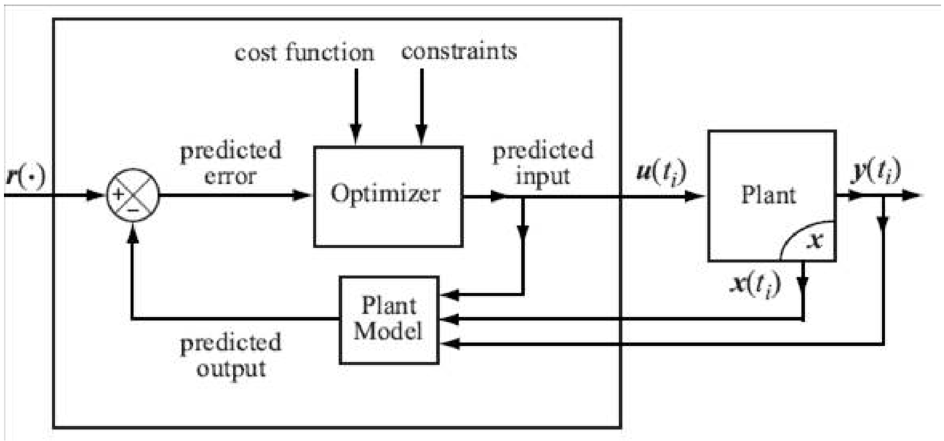

Figure 2.2 MPC structure (Zak, 2013a)

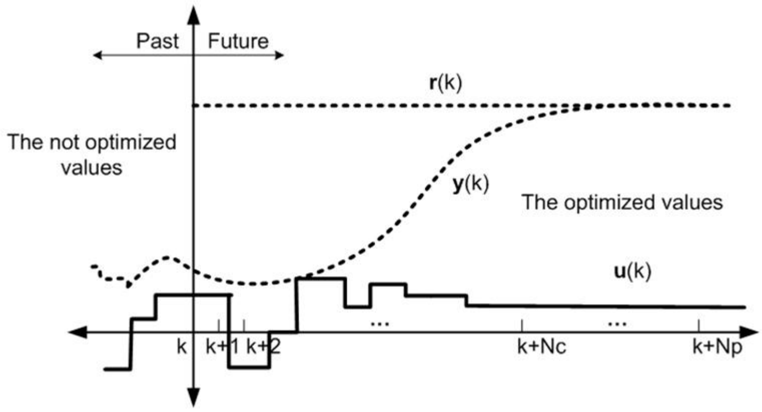

Figure 2.3 Receding horizon strategy (Boom et al., 2007)

(MIMO),

- Constraints with inputs, input rate, states and outputs.

We will be interested in deterministic MIMO linear and nonlinear time invariant dynamical system representations of process or plant.

In this thesis, we want to control deterministic linear time - invariant LTI and nonlinear dynamical systems by the proposed fast MPC method, which also has properties of closed - loop feasibility and stability (Garcia et al., 1989; Bubnicki, 2005).

LTI dynamical system is given by the following state equations (Chen, 1999; Sastry, 1999)

$$
\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) \\
x(t_0) &= x_0, \quad \forall \ t \geq 0
\end{aligned}
\tag{2.2}
$$

Nonlinear dynamical system for this work is given by the following state equations

$$
\begin{aligned}
\dot{x}(t) &= f(x(t)) + \sum_{i=1}^{r}(g_i(x(t))u_i(t)), \\
y_1(t) &= h_1(x(t)), \\
&. \\
&. \\
&. \\
y_r(t) &= h_r(x(t)), \\
x(t_0) &= x_0, \quad \forall \ t \geq 0
\end{aligned}
\tag{2.3}
$$

where state variable $x(t) \in \mathbb{R}^n$, input (control or manipulated) variable $u(t) \in \mathbb{R}^r$, output variable $y(t) \in \mathbb{R}^s$ for LTI plant, output variable $y(t) \in \mathbb{R}^r$ for the nonlinear plant, the constant matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r}$, $C \in \mathbb{R}^{s\times n}$, f, g_i are smooth nonlinear vector fields, h_j is smooth nonlinear scalar function.

Consider the discrete-time form of the plant (2.2) and the feedback linearized form of the nonlinear plant (2.3) are sampled at a control sampling period T in a microprocessor.

$$
\begin{aligned}
x(k+1) &= A_d x(k) + B_d u(k) \\
y(k) &= C_d x(k) \\
x(0) &= x_0, \quad \forall \ k \geq 0
\end{aligned}
\tag{2.4}
$$

where $k := kT$ for time instant $k = 0, 1, 2, ..., u(kT) := u(k)$ and

The equation (2.4) will be used as *the plant models in the MPC problem* to be proposed in this thesis.

$$A_d = e^{AT}, B_d = \int_0^T e^{A\tau} d\tau, C = C_d \tag{2.5}$$

Assumption 2.1. *The plants (2.2), (2.3) and (2.4) are controllable (i.e. stabilizable) by the control law $u = \mu(x)$ and the plant (2.3) has unique solution.*

Definition 14 (Plant and Plant Model)**.** *We define the dynamical systems (2.2) and (2.3) to be controlled by an MPC controller as Plant and the mathematical models of the plants used in the controller as Plant Model (2.4).*

Definition 15 (Positively Invariant (PI) Set)**.** *A set $\mathcal{X}_{pi} \subseteq \mathbb{R}^n$ is a positively invariant (PI) set of the plant model (2.4) if $A_d x_{pi}(k) + B_d\mu(x_{pi}(k) \in \mathcal{X}_{pi}$ for all $x_{pi}(k) \in \mathcal{X}_{pi}$.*

Definition 16 (Input Admissible Set)**.** *Let $u = \mu(x_{pi})$ be a control law of the system (2.4) and $\mathcal{X}_{pi} \subseteq \mathbb{R}^n$ be a subset of states. The set*

$\mathcal{X}^{\mu} = \{x \mid x \in \mathcal{X}_{pi}, \mu(x) \in \mathcal{U}\}$ is called the input admissible set for $\mathcal{X}_{pi}$.

Definition 17 (State Variable)**.** *The state x_0 of a system at time t_0 is the information at t_0 that, together with the input $u(t)$, for $t \geq t_0$, determines uniquely the output $y(t)$ for all $t \geq t_0$.*

Definition 18 (Controllability)**.** *The state equations (2.2) and (2.3) (or the pairs (A, B) and (A_d, B_d)) and (2.4) are referred to be controllable if for any intial state $x(t_0) = x_0$ and any final state $x(t_f) = x_f$, there exists an input that transfers x_0 that to x_f in a finite time. Otherwise they referred to be uncontrollable.*

Definition 19 (Observability)**.** *The state equations (2.2) and (2.4) (or the pairs (A, B) and (A_d, B_d)) and (2.3) are referred to be observable if for any unknown initial state $x(0) = x_0$, there exists a finite $t_f > 0$ such that the knowledge of the input and output $y(t)$ over $[0, t_f]$ suffices to determine uniquely the initial state x_0 . Otherwise, the equation is referred to be unobservable.*

Definition 20 (Equilibrium Point)**.** *$x_{eq} = 0$ is defined as an equilibrium point if $x(k + 1) = A_d x_{eq} = x_{eq}$ for the system (2.4) with $u = 0$ for all k.*

Theorem 2.1 (Stability in the Sense of Lyapunov)**.** *$x_{eq} = 0$ is said to be a stable equilibrium point of the plant (2.4) with $u = 0$ for all $k \geq 0$ and $\in> 0$, there exists $\delta(0, \in)$ such that*

$$\|x_0 - x_{eq}\| < \delta(0, \epsilon) \;\Rightarrow\; \|x(k) - x_{eq}\| < \epsilon \quad \forall\, k \in \mathbb{N}. \tag{2.6}$$

It can be expressed from the theorem that if the trajectory of the solution $x(k)$ starts from a initial state x_0, it ends at the the equilibrium point $x_{eq} =$, or if it starts from x_{eq}, remains at this point. Note that the equilibrium x_{eq} for nonlinear dynamical systems can be either stable or unstable functions with oscillation such as limit cycle or center instead a point. The theory is illustrated in Figure 2.4.

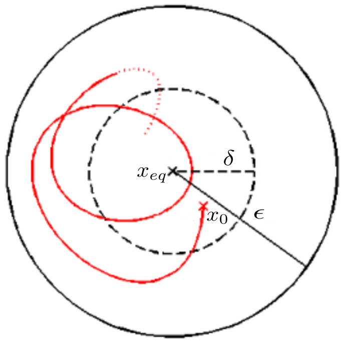

Figure 2.4 Stability in the sense of Lyapunov

The theorem (2.1) results the following stability types:

Definition 21 (Asymptotic Stability). *$x_{eq} = 0$ is said to be an asymptotically stable equilibrium point of the plant (2.4) with $u = 0$ for all $k \geq 0$ in $\mathcal{X}_{pi}$, there exists $\delta(0, \in) > 0$ such that*

$$\|x_0 - x_{eq}\| < \delta(0, \epsilon) \;\Rightarrow\; \lim_{k \to \infty} \|x(k) - x_{eq}\| = \epsilon = 0 \quad \forall\, k \in \mathbb{N}. \tag{2.7}$$

Definition 22 (Exponential Stability). *$x_{eq} = 0$ is said to be an exponential stable equilibrium point of the plant (2.4) with $u = 0$ for all $k \geq 0$ in $\mathcal{X}_{pi}$, there exists $\delta(0, \in) > 0$ such that*

$$\|x_0 - x_{eq}\| < \delta(0, \epsilon) \;\Rightarrow\; \|x(k) - x_{eq}\| \leq \sigma e^{-\rho k} \|x_0 - x_{eq}\| \quad \forall\, k \in \mathbb{N}. \tag{2.8}$$

where $\sigma, \rho > 0$.

There is the second method of Lyapunov based on the theorem (2.1). This analysis is done by measuring of energy dissipation of the energy functions defined over the trajectory $x(k)$ and time k. Using the second method of Lyapunov to determine zero – input stability of the equation (2.1), we directly use the following theorem:

Theorem 2.2 (Stability Theorem of Lyapunov)**.** *If there exists a continuously differentiable non - negative scalar function $V(x)$ and class KR function $\beta(x) \to 0$ as $k \to \infty$ with $V(0) = 0$ such that*

- *$V(x)$ is positive definite,*
- *$V(A_d x_{pi} + B_d \mu(x_{pi})) - V(x) \leq -\beta(x) \leq 0$*

whenever $\|x\|$ is sufficiently small, then $x_{eq} = 0$ is an asymptotically stable equilibrium point.

Typically, for the matrix $P > 0$, the energy function for LTI dynamical systems is chosen as $\upsilon(x(t), t) = \|x(t)\|_P^2$. The equilibrium point for LTI dynamical systems is $x_{eq} = 0$

2.3 Nonlinear Optimization

Nonlinear optimization or programming is to determine minima or maxima points of an optimization problem, which has nonlinear cost or objective. The optimization problems may be subject to equality or inequalities constraints and solved by using numerical methods (such as active set and interior points methods). The general nonliner optimization problem is given below:

Problem 2.1 (Nonlinear Programming)**.**

$$
\begin{aligned}
f(u^*) = \min_u \; & f(u) \\
& subject\,to \\
& g(u) \leq 0 \\
& h(u) = 0
\end{aligned}
\tag{2.9}
$$

where $u \in \mathbb{R}^{n_u}$ is the optimization variable, , $f(u) : \mathbb{R}^n \mapsto \mathbb{R}$ is the objective or cost function, the scalar and convex functions $g_i(u), h_j(u) : \mathbb{R}^{n_u} \to \mathbb{R}$ for $i \in \mathcal{I} =$

$\{1, ...m\}$ *and* $j \in \mathcal{E} = \{1, ...n\}$ *are the set of inequality and equality constraints, respectively.*

Assumption 2.2. f *is twice - differentiable at* u *and has continuous second derivatives. In this case, it can be approximate function in the neighbourhood of an arbitrary point* u *point using the following Taylor series*

$$f(u+\Delta u) \approx f(u)+\nabla f(u)(u+\Delta u-u)+\frac{1}{2}(u+\Delta u-u)^T\nabla^2 f(u)(u+\Delta u-u)+h.o.t \tag{2.10}$$

where $\nabla f(u)$ *is the first - order derivative or gradient component of* $f(u + \Delta u)$ *and* $\nabla^2 f(u) \triangleq H(u)$ *is the second - order derivative or hessian component of* $f(u + \Delta u)$*,* $h.o.t$ *is the other high - order terms.*

Assumption 2.3. *Suppose* u^* *is the feasible local or global optimal (minimum) value and minimizer of* $f(u)$ *and* $f(u^*)$ *is the minimum cost value.*

2.3.1 Unconstrained Optimization

If the problem (2.9) is not subject to the constraints, it is the unconstrained optimization problem. If the problem is not convex, its solution is a local minimum. The necessary and sufficient conditions for a local or global minimum are determined by the following theorems:

Theorem 2.3 (Necessary Condition for a Local Minimum)**.** *If* u^* *is a local minimum, then* $\nabla f(u^*) = 0$.

Theorem 2.4 (Sufficient Condition for a Strict Local Minimum)**.** u^* *is a strict local minimum if* $\nabla f(u^*) = 0$ *and* $H(u) > 0$ *(positive definite).*

Theorem 2.5 (Necessary and Sufficient Conditions for a Global Minimum)**.** *The local minimum* u^* *of* f *is also a global minimum if* f *is defined over a convex set* $\mathcal{X}$.

The unconstrained optimization problems are solved the following

- the algorithms, which ends in a finite number of steps,

- the iterative methods, which evaluate gradient (i.e. gradient descent methods) and hessian (i.e. newton methods) of the Taylor expansion of the unconstrained cost,
- heuristic methods, which may satisfy approximate solutions,

2.3.2 Constrained Optimization

We focus the convex optimization problems. The convex optimization for global optimal control law is an important tool in the applications of optimal control theory.

Assumption 2.4. *In the optimization problem (2.9), f and g are defined over a convex set $\mathcal{X}$ and h has affine function.*

Problem 2.2 (Dual Convex Optimization)**.**

$$\begin{aligned} f_d(\upsilon^*, \lambda^*) &= \max_{\upsilon,\lambda}\ f_d(\upsilon, \lambda) \\ &subject\,to \\ &\lambda \geq 0 \end{aligned} \tag{2.11}$$

where $f_d(\upsilon, \lambda) = min_u \mathcal{L}(u, \upsilon, \lambda)$, $\lambda \in \mathbb{R}^m$, $\upsilon \in \mathbb{R}^n$ are the Lagrange Multipliers associated with the inequality and equality constraints, respectively, and $\mathcal{L}$ is the Lagrange Function (or Lagrangian) given by

$$\mathcal{L}(u, \upsilon, \lambda) = f(u) + \lambda^T g(u) + \upsilon^T h(u) \tag{2.12}$$

The difference between a primal feasible and a dual feasible solution is called the duality gap.

Definition 23 (Slater's condition)**.** *It is assumed that the primal problem (2.9) is strictly feasible and there exists a u such that $g(u) < 0$ and $h(u) = 0$. This hold duality.*

Theorem 2.6 (Karush-Khun-Tucker (KKT) Optimality Conditions)**.** *Let u be feasible point for the convex optimization problem 2.9 and $g_i(u)$ and $h_j(u)$ be continuously* *differentiable. If u^* is a global minimum, there exists the vectors υ^* and λ^* such that the following conditions are satisfied by*

$$\nabla_u \mathcal{L}(u^*, \upsilon^*, \lambda^*) = 0, \tag{2.13}$$

$$g(u^*) \leq 0, \tag{2.14}$$

$$h(u^*) = 0, \tag{2.15}$$

$$\lambda^* \geq 0, \tag{2.16}$$

$$\lambda^* g(u^*) = 0, \;\; \forall i \tag{2.17}$$

Note that $g(.)$ and $h(.)$ must be twice differentiable to obtain sufficiency conditions for a local minimum or minima (Bertsekas, 2005). In the KKT conditions or system, the inequalities (2.14-2.16) are the primal and dual feasibility conditions, the equality (2.13) is the minimization of $\mathcal{L}$ and the equality (2.17) is the complementary condition. The KKT conditions satisfy necessary and sufficient conditions for the global minimum or minima in the convex optimization.

If the cost function and constraints of optimization problem (2.9) are convex (or quadratic), the problem is called *convex quadratic program (QP)*.

The inequality constrained optimization problems can be iteratively and effectively solved by

- the active set methods,
- the interior point methods.

Problem 2.3 (Convex Quadratic Optimization)**.**

$$\begin{aligned} f^*(u^*) = \min_u \; & u^T F u + c^T u \\ & subject\, to \\ & Gu \leq b_g \\ & Hu = b_{\mathrm{g}} \end{aligned} \tag{2.18}$$

where $F \in \mathbb{R}^{n_u \times n_u}$ and if $F \geq 0$, the problem (2.9) is the strictly convex and minimized over a polyhedron $\mathcal{P}$.

2.3.2.1 Active - Set Methods

Definition 24 (Active Set)**.** *Let u be a feasible. The inequality constraint is defined as active set if $g_i(u) = 0$ at u. The active set $\mathcal{A}(u)$ is given by*

$$\mathcal{A}(u) = \mathcal{E} \cup \{i \in \mathcal{I} \mid g_i(u) = 0\} \tag{2.19}$$

The optimal solution u^* can be iteratively found the following equality constrained problem corresponding the problem (2.9).

Problem 2.4 (Equality Constrained Optimization)**.**

$$\begin{gathered} f(u^*) = \min_u \; f(u) \\ subject\, to \\ \mathcal{A}(u) = 0 \end{gathered} \tag{2.20}$$

or the following KKT system of the problem (2.20) the corresponding equations (2.13) and (2.15) of the KKT system,

$$\nabla_u \mathcal{L}(u^*, \upsilon^*) = 0 \tag{2.21}$$

$$\mathcal{A}(u^*) = 0 \tag{2.22}$$

The equalities (2.21-2.22) are solved by

- converting inequality constraints to equality constraints by using active sets (active set methods),
- evaluating the obtained unconstrained optimization problem eliminating affine equality constraints,

- representing equality constraints as inequality constraints (i.e. interior point methods)

2.3.2.2 *Newton's Method*

$f(u + \Delta u)$ is approximated into the second-order Taylor expansion (quadratic expansion) and $\mathcal{A}(u+\Delta u)$ is approximated into the first-order Taylor expansion about a feasible u and a minimum step (search direction, Newton direction or Newton step) Δu is computed by the following optimization problem

Problem 2.5 (Equality Constrained Optimization based on Newton's Method)**.**

$$\begin{aligned}&\min_{u}\ f(u+\Delta u) \approx f(u) + \nabla f(u)\Delta u + \Delta u^T \nabla^2 f(u)\Delta u \\ &\textit{subject to} \\ &\mathcal{A}(u+\Delta u) \approx \mathcal{A}(u) + \nabla \mathcal{A}(u)\Delta u = 0\end{aligned} \tag{2.23}$$

such that

- f must be decreased as $f(u+\Delta u) \leq f(u)$
- The feasible solution must be satisfied on the equality constrained $\mathcal{A}(u+\Delta u) = 0$

The lagrange function is

$$\mathcal{L}(\Delta u, \upsilon) = f(u) + \nabla f(u)\Delta u + \Delta u^T \nabla^2 f(u)\Delta u + \upsilon^T(\mathcal{A}(u) + \nabla \mathcal{A}(u)\Delta u) \tag{2.24}$$

The KKT conditions are

$$\nabla_{\Delta u}\mathcal{L}(\Delta u^*, \upsilon^*) = 0 \tag{2.25}$$

$$\mathcal{A}(u^*) + \nabla \mathcal{A}(u^*)\Delta u^* = 0 \tag{2.26}$$

and the KKT system is

$$\begin{bmatrix} \nabla^2 f(u^*) & \nabla \mathcal{A}^T(u^*) \\ \nabla \mathcal{A}(u^*) & 0 \end{bmatrix} \begin{bmatrix} \Delta u^* \\ \upsilon^* \end{bmatrix} = \begin{bmatrix} -\nabla f(u^*) \\ -\mathcal{A}(u^*) \end{bmatrix} \tag{2.27}$$

The system can be iteratively solved by

$$\begin{bmatrix} \nabla^2 f(u(k)) & \nabla \mathcal{A}^T(u(k)) \\ \nabla \mathcal{A}(u(k)) & 0 \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \upsilon(k) \end{bmatrix} = \begin{bmatrix} -\nabla f(u(k)) \\ -\mathcal{A}(u(k)) \end{bmatrix} \quad \forall k \tag{2.28}$$

and

$$u(k+1) = u(k) + \gamma(k)\Delta u(k) \tag{2.29}$$

where $\gamma(k)$ is step size or length.

The optimal value of $\gamma(k)$ can be determined by

- *Backtracking line-search* of $f(u(k) + \gamma(k)\Delta u(k))$
- *Filter methods.*

The Newton's method and the backtracking line search algorithms are given in Algorithms 1 and 2.

Algorithm 1 Newton's method

1: Given a feasible $u(k)$, set $k = 0$
2: Compute $\Delta u(k)$
3: Jump Backtracking Line Search algorithm for optimal γ
4: Set $u(k+1) = u(k) + \gamma(k)\Delta u(k)$ and $k = k+1$ go to step 1

Algorithm 2 Backtracking line search algorithm

1: Given a feasible u, descent direction Δu and the parameters $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$
2: Set $\gamma := 1$
3: While $f(u(k) + \gamma \Delta u(k)) > f(u(k)) + \alpha\gamma\nabla f(u(k))^T \Delta u(k)$, $\gamma := \beta\gamma$

It is possible to reach the optimal solution in an optimization problem by using the active set methods with numerous iterations; however, the computation load at each iteration is low - level than the interior point methods. The active sets can be applied to the sparse formulation of the optimization problem and thus the size of the problem and the computation load can be reduced at each iteration. Since the problem has active sets, an initial warm-start enables the problem to reach a quick solution.

2.3.2.3 Interior - Point Methods

Interior point or barrier methods are used for solving linear and nonlinear convex optimization problems. In this approach, a barrier function is defined in the optimization problem (2.9) and the problem is converted to the equality constrained problems and then can be computed the optimal solution by using Newton's method.

Problem 2.6 (Barrier Method)**.**

$$\begin{aligned} f(u^*) = \min_u \ & f(u) + \omega\rho(z) \\ & subject\,to \\ & g(u) + z = 0 \\ & h(u) = 0 \\ & -g(u), z \geq 0 \end{aligned} \tag{2.30}$$

where $z \in \mathbb{R}^m$ is the slack variable, $\omega > 0$ is a small barrier parameter and the barrier function

$$\rho(z) = -\sum_{i=0}^{m} log(-z_i) \tag{2.31}$$

The problem (2.30) is rewritten into the following equality constrained problem

$$\begin{aligned} f(u^*) = \min_u \ & f(u) - \omega\rho(-g(u)) \\ & subject\,to \\ & h(u) = 0 \\ & -g(u) \geq 0 \end{aligned} \tag{2.32}$$

The lagrangian function is written as

$$L(u, \upsilon, \omega) = f(u) - \omega\rho(-g(u)) + \upsilon^T h(u) \tag{2.33}$$

and the necessary optimality condition is given by

$$\nabla_u L(u, \upsilon, \lambda) = \nabla f(u) + \nabla g(u)(-\frac{\omega}{g(u)}) + \nabla h(u)^T \upsilon \tag{2.34}$$

where $\nabla g(u) = [\nabla g_1(u) \ldots \nabla g_m(u)]^T$.

Let $\lambda \triangleq -\frac{\omega}{g(u)}$ be then the perturbed KKT conditions for the logarithmic barrier problem are given by

$$\nabla_u L(u, \upsilon, \lambda) = 0, \tag{2.35}$$

$$\boldsymbol{\lambda} g(u) = -\omega \mathbf{1}, \tag{2.36}$$

$$h(u) = 0, \tag{2.37}$$

$$\lambda, -g(u) \geq 0, \quad \forall i \tag{2.38}$$

where $\boldsymbol{\lambda} = Diag(\lambda)$ and $\mathbf{1} = Diag(1)$.

The these conditions are equated to zero by the following definition

$$\kappa(u, \upsilon, \lambda) \triangleq \begin{bmatrix} \nabla f(u) + \nabla g(u)\lambda + \nabla h(u)^T \upsilon \\ \boldsymbol{\lambda} g(u) + \omega \mathbf{1} \\ h(u) \end{bmatrix} = 0 \tag{2.39}$$

$\kappa(u+\Delta u, \upsilon+\Delta\upsilon, \lambda+\Delta\lambda)$ is approximated into the first-order Taylor expansion about a feasible vector $\begin{bmatrix} u & \upsilon & \lambda \end{bmatrix}^T$ and a minimum search direction vector $\begin{bmatrix} \Delta u & \Delta \upsilon & \Delta\lambda \end{bmatrix}^T$ are computed.

$$\kappa(u + \Delta u, \upsilon + \Delta\upsilon, \lambda + \Delta\lambda) \approx \kappa(u, \upsilon, \lambda) + \nabla\kappa(u, \upsilon, \lambda) \begin{bmatrix} \Delta u & \Delta \upsilon & \Delta\lambda \end{bmatrix}^T = \mathbf{0} \tag{2.40}$$

As $\omega \to 0$, the solution converges to the optimal solution of (2.9) very close by updating

$$\begin{aligned} u(k+1) &= u(k) + \gamma(k)\Delta u(k) \\ \upsilon(k+1) &= \upsilon(k) + \gamma(k)\Delta \upsilon(k) \\ \lambda(k+1) &= \lambda(k) + \gamma(k)\Delta\lambda(k) \end{aligned} \tag{2.41}$$

In an optimization problem using the interior point methods, the optimal solution is achieved through a small number of iterations; however, in these iterations the computation load is high - level and this computational complexity grows polynomially. If it is possible, the size of the optimization problem can be reduced decreasing the computational load and optimization can be started with an initial warm-start solution.

CHAPTER 3

UPSAMPLED MODEL PREDICTIVE CONTROL FOR LINEAR - TIME INVARIANT DYNAMICAL SYSTEMS

Model Predictive Control is an constrained, online and open - loop optimal control method based on current and future prediction computations of prediction plant model, which is embedded in a digital computer, over finite receding (or moving) time horizon (or window) and on physical constraints of plant. It is the most advanced version of the optimal control methods because of its online computation strategy over receding time horizon window. The other optimal control methods outlined in Chapter 1 have the precomputed control law on plant model over fixed time horizon (such as LQR over infinity time horizon). The computation strategy results in computation of repeated, updated and more effective nonlinear feedback control law vectors.

In MPC strategy, computed control law vectors compensate mismatches between plant, plant model and external (or unforeseen) disturbances on constraints according to changing new conditions at each optimization sampling period and the first components of the vectors are implemented to plant until reaching desired output point or function (Diehl & Gros, 2017).

Although MPC is theoretically well established, in practice applications it is not applicable but there are some methods with significant limits for fast plants. These methods are based on the direct optimization methods and they are more convenient method for slow plants under the online computation load given detailed in Chapter 1.

In this chapter, we propose a method for linear dynamical systems to speed the slow response issues (rise time, settling time and prediction) of the existing MPC algorithms up due to the computation load of online optimization without reducing long computation time. In the method, original MPC strategy is warped using an integer time scaling factor in time domain. The aim is to speed this strategy up by exploiting the speeds of fast microprocessors. For this aim an integer time scaling factor is selected according to the limit of the used microprocessor. Solution of plant, which is controlled, is contracted and sampled by this scaling factor. The nominal MPC problem is based on this modification of plant. The first elements of fast computed control law vectors are dilated until original online optimization sampling periods is reached again and applied to plant at each original optimizaiton sampling period. The modified MPC strategy, which is called *Upsampled MPC (UMPC)*, settles faster to desired output than original MPC strategy and also satisfies faster prediction at each sampling period. Since the method is more compatible with fast microprocessor and the speed of UMPC results from microprocessor. Thus, the slow operation issue of MPC algorithms is solved for the different time scaling factors using this method. The original MPC strategy is contracted in time, the control energy computed in UMPC strategy is lower than the original MPC. There also exists the free times between online optimization sampling periods of original MPC and UMPC strategies. These free times can be used for improvement of the existing MPC algorithms. The proposed approach is applied on linearized Citation Cessna aircraft and the results are compared to the original MPC.

3.1 Time - Scaled and Upsampled Solutions of LTI Dynamical Systems

Consider the deterministic Multi-Input Multi-Output (MIMO) LTI plant is given by the following state equation (Chen, 1999)

$$\begin{aligned} \frac{dx_p(t)}{dt} &= Ax_p(t) + Bu_p(t) \\ y_p(t) &= Cx_p(t) \\ x_p(t_0) &= x_0, \quad \forall \ t \geq t_0 \end{aligned} \tag{3.1}$$

where state variable $x_p(t) \in \mathbb{R}^n$, initial state variable $x_0 \in \mathbb{R}^n$, input/control variable $u_p(t) \in \mathbb{R}^r$, output variable $y_p(t) \in \mathbb{R}^s$, constant matrices $A \in \mathbb{R}^{n\times n}$,$B \in \mathbb{R}^{n\times r}$, $C \in \mathbb{R}^{s\times n}$ of the plant (3.1), and continuous time variable $t \in \mathbb{R}^+$. Assume the plant is controllable.

The solution x_p for the plant (3.1) is given by

$$\begin{aligned} x_p(t) &= e^{A(t-t_0)}x_0 + \int_{t_0}^{t} e^{A(t-\tau)}Bu_p(\tau)d\tau \\ &\triangleq x_{px}(t) + x_{pu}(t) \end{aligned} \tag{3.2}$$

$x_p(t)$ is a linear combination of $x_{px}(t)$ and $x_{pu}(t)$ resulting from superposition theorem. $x_{px}(t)$ is zero-input response with non-zero initial state x_0 and $x_{pu}(t)$ is zero-state with non - zero input.

Consider a microprocessor, which can operate faster than online optimization sampling period of the Original MPC, and under this condition, we can select a large enough integer time scaling factor $M > 1$. Then we can define the following dynamics for the time - scaled solutions of plant (3.1)

$$\begin{aligned} \frac{dx_m(t)}{dt} &= M(Ax_m(t) + Bu_m(t)) \\ y_m(t) &= Cx_m(t) \\ x_m(t_0) &= x_0, \quad \forall\ t \geq t_0 \end{aligned} \tag{3.3}$$

where state variable $x_m(t) \in \mathbb{R}^n$, the measured or estimated initial state variable $x_0 \in \mathbb{R}^n$ from the plant (3.1), input/control variable $u_m(t) \in \mathbb{R}^r$, output variable $y_m(t) \in \mathbb{R}^s$ of the dynamic (3.3). The time-scaled state space dynamic is called *contracted plant* in time domain. Multiplying e^{-MAt} on both side of state equation of the contracted plant (3.3), the following equation is yielded

$$e^{-MAt}\frac{dx_m(t)}{dt} - Me^{-MAt}Ax_m(t) = Me^{-MAt}Bu_m(t) \tag{3.4}$$

This equation is the same to the following equation

$$\frac{d}{dt}(e^{-MAt}x_m(t)) = Me^{-MAt}Bu_m(t) \tag{3.5}$$

Integrating both side of the equation from t_0 to t

$$e^{-MA\tau}x_m(\tau)\big|_{\tau=t_0}^{t} = M\int_{t_0}^{t} e^{-MA\tau}Bu_m(\tau)d\tau \tag{3.6}$$

we reach

$$e^{-MAt}x_m(t) - e^{-MAt_0}x_0 = M\int_{t_0}^{t} e^{-MA\tau}Bu_m(\tau)d\tau \tag{3.7}$$

Thus we have the solution $x_m(t)$ of the contracted plant

$$x_m(t) = x_0 e^{MA(t-t_0)} + M\int_{t_0}^{t} e^{MA(t-\tau)} B u_m(\tau) d\tau \tag{3.8}$$

Using change of $t = Mt'$, $t_0 = Mt'_0$ and $\tau = M\tau'$ on $x_p(t)$

$$\underbrace{x_p(t)}_{t=Mt'} = \underbrace{x_0 e^{A(t-t_0)}}_{t=Mt',t_0=Mt'_0} + \underbrace{e^{At}}_{t=Mt'}\int_{t_0}^{t}\underbrace{e^{-A\tau} B u_p(\tau) d\tau}_{\tau=M\tau'} \tag{3.9}$$

we can write the time scaling relation between the solutions of the plant $x_p(t)$ and the contracted plant $x_m(t)$

$$x_m(t) = x_p(Mt) \tag{3.10}$$

under the condition

$$u_m(t) = u_p(Mt) \tag{3.11}$$

$x_m(t)$ has faster response by M times than $x_p(t)$ in time domain.

Consider the discrete-time form of the plant (3.1) is sampled at the online optimization sampling periods kT_p of the original MPC algorithm.

$$\begin{aligned} x_p(k+1) &= A_{pd} x_p(k) + B_{pd} u_p(k) \\ y_p(k) &= C x_p(k) \\ x_p(0) &= x_0, \quad \forall \ k \geq 0 \end{aligned} \tag{3.12}$$

where $k := kT_p$ for time instant $k = 0, 1, 2, ...$, $u_p(kT_p) := u_p(k)$ and

$$\begin{aligned} A_{pd} &= e^{AT_p} \\ B_{pd} &= (\int_0^{T_p} e^{A\tau} d\tau) B \end{aligned} \tag{3.13}$$

The solution is given by

$$
\begin{aligned}
x_p(k) &= A_{pd}^k x_0 + \sum_{m=0}^{k-1} A_{pd}^{k-1-m} B_{pd} u_p(m) \\
&\triangleq x_{px}(k) + x_{pu}(k)
\end{aligned}
\tag{3.14}
$$

The discrete-time plant (3.12) with T_p is the plant model used in the original MPC algorithm with T_p and can also be evaluated as the plant instead the continuous - time plant (3.1) controlled by the same controller.

We claim that the original MPC algorithm is contracted by M and has a faster online optimization period $T_m = T_p/M$. This algorithm will be called *UMPC*. For this aim the contracted plant (3.3) is sampled by $T_m = T_p/M$.

Since

$$
\frac{dx_m(t)}{dt} = \lim_{T_m \to 0} \frac{x_m(t+T_m) - x_m(t)}{T_m} \tag{3.15}
$$

the approximation of the plant (3.3) is written as

$$
x_m(t+T_m) = x_m(t) + MT_m(Ax_m(t) + Bu_m(t)) \tag{3.16}
$$

If x_m and y_m are computed at $t = kT_m$ for $k = 0, 1, ...$, then (3.3) becomes

$$
\begin{aligned}
x_m((k+1)T_m) &= (I + MAT_m)x_m(kT_m) + MT_m Bu_m(kT_m) \\
y_m(kT_m) &= Cx_m(kT_m) \\
x_m(0) &= x_0, \quad \forall \ k \geq 0
\end{aligned}
\tag{3.17}
$$

Let the control variable be

$$
u_m(t) = u_m(kT_m) =: u_m(k) \quad for \ \ kT_m \leq t \leq (k+1)T_m \tag{3.18}
$$

for $k = 0, 1, ...$

Computing the solution of the plant model (3.8) at kT_m, we yield the following the solution

$$x_m(k) := x_m(kT_m) = e^{MAkT_m}x_0 + M\int_0^{kT_m} e^{MA(kT_m-\tau)}Bu_m(\tau)d\tau \tag{3.19}$$

The solution at $(k+1)T_m$ is computed as

$$x_m(k+1) := x_m[(k+1)T_m] = e^{MA(k+1)T_m}x_0 + M\int_0^{(k+1)T_m} e^{MA((k+1)T_m-\tau)}Bu_m(\tau)d\tau \tag{3.20}$$

If the equation (3.20) is rearranged, we can write it as

$$\begin{aligned} x_m(k+1) = e^{MAkT_m}[e^{MAkT_m}x_0 + M\int_0^{kT_m} e^{MA(kT_m-\tau)}Bu_m(\tau)d\tau] \\ + \int_{kT_m}^{(k+1)T_m} e^{MA(kT_m+T_m-\tau)}Bu_m(\tau)d\tau \end{aligned} \tag{3.21}$$

If a new variable $\beta := kT_m + T_m - \tau$ is defined, the equation (3.21) becomes

$$x_m(k+1) = e^{MAT_m}x_m(k) + M(\int_0^{T_m} e^{MA\beta}d\beta)Bu_m(k) \tag{3.22}$$

If it is assumed the control variable changes only at time instants kT_m and if the plant model (3.3) is sampled at kT_m, the following discrete-time form is yielded

$$\begin{aligned} x_m(k+1) &= A_{md}x_m(k) + B_{md}u_m(k) \\ y_m(k) &= Cx_m(k) \\ x_m(0) &= x_0, \quad \forall\ k \geq 0 \end{aligned} \tag{3.23}$$

where $k := kT_m$ for time instant $k = 0, 1, ..., u_m(kT_m) := u_m(k)$ and

$$\begin{aligned} A_{md} &= e^{MAT_m} = A_{pd} \\ B_{md} &= M(\int_0^{T_m} e^{MA\tau}d\tau)B = B_{pd} \end{aligned} \tag{3.24}$$

Note that $B_{md} = B_{pd}$ can be yielded using change of variables.

B_{md} can be approximated into power series as

$$\begin{aligned} B_{md} &= M(\int_0^{T_m} (I + MA\tau + (MA)^2\frac{\tau^2}{2!} + ...)d\tau \\ &= M(T_m I + \frac{T_m^2}{2!}MA + \frac{T_m^3}{3!}(MA)^2 + \frac{T_m^4}{4!}(MA)^3 + ...) \end{aligned} \tag{3.25}$$

using the Taylor series

$$e^{MAt} = I + tMA + \frac{t^2}{2!}(MA)^2 ... = \sum_{k=0}^{\infty} \frac{1}{k!}(MA)^k \tag{3.26}$$

If A is singular, the series (3.25) is written as

$$(MA)^{-1}(MT_m A + \frac{T_m^2}{2!}(MA)^2 + \frac{T_m^3}{3!}(MA)^3 + ... + I - I) = (MA)^{-1}(e^{MAT_m} - I) \tag{3.27}$$

using the series (3.26). Thus, the matrix B_{md} can be used

$$B_{md} = A^{-1}(A_{md} - I)B \tag{3.28}$$

instead of computing the infinite series (3.26).

The solution is given by

$$\begin{aligned} x_m(k) &= A_{md}^k x_0 + \sum_{m=0}^{k-1} A_{md}^{k-1-m} B_{md} u_m(m) \\ &\triangleq x_{mx}(k) + x_{mu}(k) \end{aligned} \tag{3.29}$$

The state space equation (3.23) is called *Upsampled Plant Model (UPM)* of UMPC.

The dynamic pairs (3.12) and (3.23) have the same behavior at different sampling periods. We can write the following time-shifted relationship between $x_p(k)$ and $x_m(k)$.

$$x_p(kT_p) = x_m(k(T_m + T_{free}))\ \ \forall\, k \tag{3.30}$$

where

$$\begin{aligned} T_{free} &= T_p - T_m \\ &= T_m(M-1) \end{aligned} \tag{3.31}$$

We can state from the equation (3.30) that if the values of $x_m(kT_m)$ are dilated by T_{free} and storing its values at each time instant k, it is equated to $x_p(kT_p)$. In this case, the UPM (3.23) is transformed to the original plant model (3.12).

3.1.1 Case Study

3.1.1.1 Case 1

Consider the feedback linearized simple pendulum

$$\begin{aligned} \dot{x}_{p1}(t) &= x_{p2}(t) \\ \dot{x}_{p2}(t) &= u_p(t) \\ x_p(0) = x_0 &= [x_{01}\ x_{02}]^T, \quad \forall\ t \geq 0 \end{aligned} \tag{3.32}$$

where $x_1(t)$ is rad/sn.

For $u_p(t) = sin(t)$, the solution is

$$\begin{aligned} x_{p1}(t) &= x_{01} + (x_{02}+1)t - sin(t) \\ x_{p2}(t) &= x_{02} - cos(t) + 1 \end{aligned} \tag{3.33}$$

The contracted plant is

$$
\begin{aligned}
\dot{x}_{m1}(t) &= Mx_{m2}(t) \\
\dot{x}_{m2}(t) &= Mu_m(t) \\
x_m(0) &= x_0 = [x_{01}\ x_{02}]^T, \quad \forall\ t \geq 0
\end{aligned} \tag{3.34}
$$

For $u_p(t)(t) = sin(t)$, the solution is

$$
\begin{aligned}
x_{m1}(t) &= x_{01} + (x_{02} + 1)Mt - sin(Mt) \\
x_{m2}(t) &= x_{02} - cos(Mt) + 1
\end{aligned} \tag{3.35}
$$

For $u_p(t) = t + tsin(t)$, the solutions are

$$
\begin{aligned}
x_{p1}(t) &= x_{01} + x_{02}t + 1/6t^3 - tsin(t) - 2cos(t) + 2 \\
x_{p2}(t) &= x_{02} + 1/2t^2 + sin(t) - tcos(t)
\end{aligned} \tag{3.36}
$$

$$
\begin{aligned}
x_{m1}(t) &= x_{01} + x_{02}Mt + 1/6(Mt)^3 - Mtsin(Mt) \\
&\quad - 2cos(Mt) + 2 \\
x_{m2}(t) &= x_{02} + 1/2(Mt)^2 + sin(Mt) - Mtcos(Mt)
\end{aligned} \tag{3.37}
$$

The solution pairs (3.33-3.35) and (3.36-3.37) satisfy the equation (3.30) under the condition (3.31) (Wolfram Alpha LCC, 2020).

3.1.1.2 Case 2

Consider the linearized continuous-time plant (Citiation Aircraft) shown in Figure 3.1.

$$
\begin{aligned}
\dot{x}_p(t) &= \begin{bmatrix} -1.2822 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.4293 & 0 & -1.8366 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{p1}(t) \\ x_{p2}(t) \\ x_{p3}(t) \\ x_{p4}(t) \end{bmatrix} + \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u_p(t) \\
y_p(t) &= \begin{bmatrix} y_{p1}(t) \\ y_{p2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{p1}(t) \\ x_{p2}(t) \\ x_{p3}(t) \\ x_{p4}(t) \end{bmatrix} \\
x_p(0) &= x_0, \quad \forall \ t \geq 0
\end{aligned} \tag{3.38}
$$

where $u_p(t)$ is the elevator angle, $x_{p1}(t)$ is the angle of attack, $x_{p2}(t)$ is the pitch angle, $x_{p3}(t)$ is the pitch rate, $x_{p4}(t)$ is the altitude, the outputs $y_{p1}(t)$ and $y_{p2}(t)$ are the pitch angle and the altitude, respectively.

The constraints are the elevator angle $-0.262 \leq u(t) \leq 0.262\,rad.$, the elevator rate $-0.0698 \leq \Delta u_p(t) \leq 0.0698\,rad.$,the pitch angle $-0.349 \leq y_{p1}(t) \leq 0.349\,rad.$ The plant is linearized at altitude of 5000 m. and a speed of 128.2 m. / sec. The open-loop impulse response vector is unstable due to open-loop poles of the solution $x_p(t)$ are $0, 0, -1.5594 \pm 2.29j$ (M. Zeilinger & Morari, 2016).

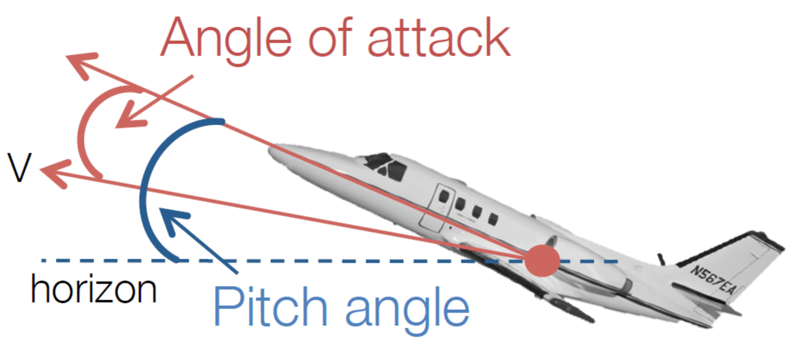

Figure 3.1 Cessna aircraft (M. Zeilinger & Morari, 2016)

The plant model of the Original MPC for the plant (3.38) at the online optimization sampling period $T_p = 0.2\ sec.$ is

$$x_p(k+1) = \begin{bmatrix} 0.6958 & 0 & 0.1385 & 0 \\ -0.08697 & 1 & 0.1619 & 0 \\ -0.7674 & 0 & 0.6174 & 0 \\ -22.68 & 25.64 & 0.2272 & 1 \end{bmatrix} \begin{bmatrix} x_{p1}(k) \\ x_{p2}(k) \\ x_{p3}(k) \\ x_{p4}(k) \end{bmatrix} + \begin{bmatrix} -0.3181 \\ -0.2952 \\ -2.726 \\ 0.4953 \end{bmatrix} u_p(k)$$

$$y_p(k) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{p1}(k) \\ x_{p2}(k) \\ x_{p3}(k) \\ x_{p4}(k) \end{bmatrix}$$

$$x_p(0) = x_0, \quad \forall \ k \geq 0 \tag{3.39}$$

The contracted plant for $M = 10$ is

$$\dot{x}_m(t) = 10 \begin{bmatrix} -1.2822 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.4293 & 0 & -1.8366 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{m1}(t) \\ x_{m2}(t) \\ x_{m3}(t) \\ x_{m4}(t) \end{bmatrix} + 10 \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u_m(t)$$

$$y_m(t) = \begin{bmatrix} y_{m1}(t) \\ y_{m2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{m1}(t) \\ x_{m2}(t) \\ x_{m3}(t) \\ x_{m4}(t) \end{bmatrix}$$

$$x_m(0) = x_0, \quad \forall \ t \geq 0 \tag{3.40}$$

where $u_m(t) = u_p(10t)$.

The UPM of UMPC at $T_m = 0.02\ sec.$ is

$$x_m(k+1) = \begin{bmatrix} 0.6958 & 0 & 0.1385 & 0 \\ -0.08697 & 1 & 0.1619 & 0 \\ -0.7674 & 0 & 0.6174 & 0 \\ -22.68 & 25.64 & 0.2272 & 1 \end{bmatrix} \begin{bmatrix} x_{m1}(k) \\ x_{m2}(k) \\ x_{m3}(k) \\ x_{m4}(k) \end{bmatrix} + \begin{bmatrix} -0.3181 \\ -0.2952 \\ -2.726 \\ 0.4953 \end{bmatrix} u_m(k)$$

$$y_m(k) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{m1}(k) \\ x_{m2}(k) \\ x_{m3}(k) \\ x_{m4}(k) \end{bmatrix}, \quad x_m(0) = x_0, \quad \forall\ k \geq 0 \tag{3.41}$$

The equations (3.39) and (3.41) satisfy the equation (3.30). The impulse responses are illustrated in Figure 3.2. As the plant model reaches a value in $5\ sec.$, the UPM reaches the same value in $0.5\ sec.$

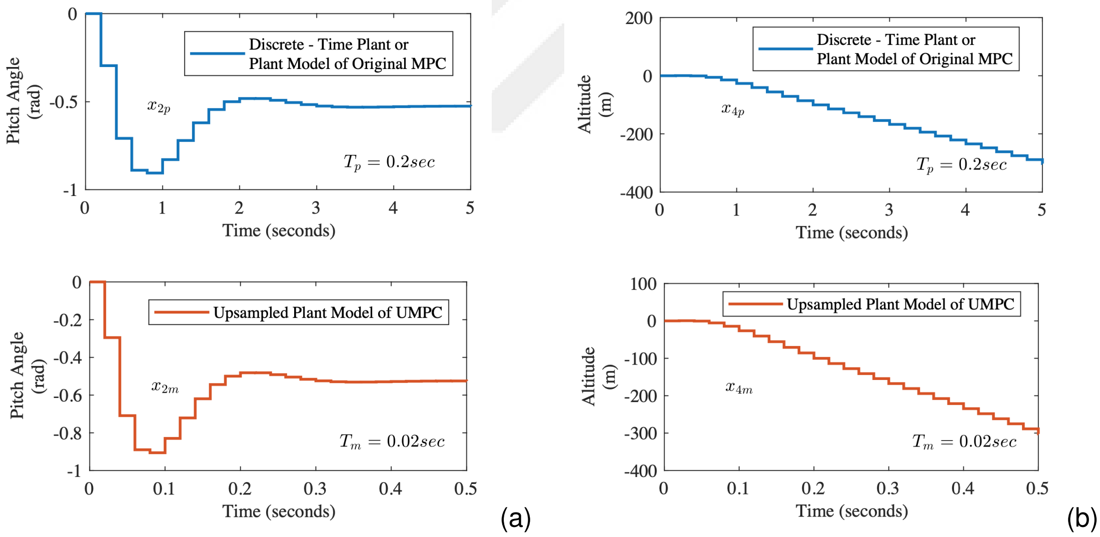

Figure 3.2 Discrete - time plant or plant model of original MPC and UPM of UMPC impulse responses (a) The pitch angle impulse responses (b) The altitude impulse responses

The free time between the plant model (3.39) of MPC and the UPM (3.41) of UMPC are $T_{free} = 0.18\ sec.$ at each $T_p = 0.2\ sec.$ If the UPM (3.41) of UMPC is dilated by $T_{free} = 0.18\ sec.$ at each $T_m = 0.02\ sec.$, it is equated to plant model (3.39) of the Original MPC with $T_p = 0.2\ sec.$

3.2 Nominal UMPC

We establish a nominal or standard predictive control algorithm based on the UPM. Assume the plant (3.1) or (3.12) must be controlled by the original MPC algorithm, which has the plant model (3.12) at the online optimization sampling period T_p and does not have an error between the plant and the plant model. We want to speed its response (rise time, settling time and prediction) up. For this aim it is formulated based on the UPM (3.23) instead of the plant model (3.12) and the online optimization sampling period becomes T_m. we call this approach as *Upsampled MPC (UMPC)*. The MPC and UMPC block diagrams are depicted in Figure 3.3.

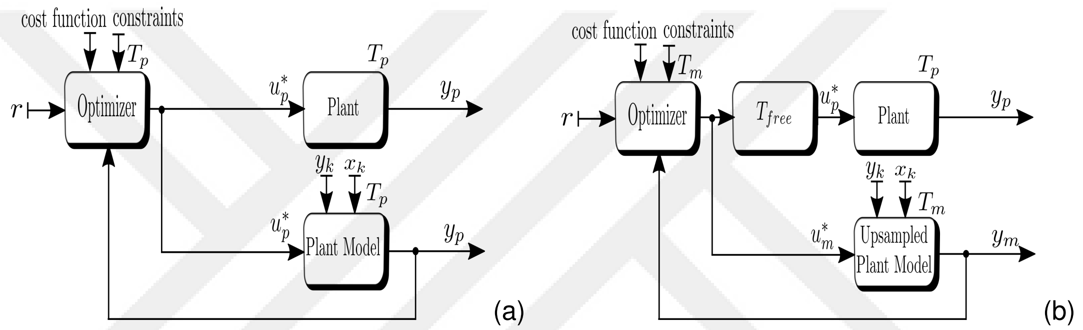

Figure 3.3 Equivalent MPC controller block structures (a) Original MPC block structure (b) UMPC block structure

The prediction UPM for the UPM (3.23) with T_m is

$$\begin{aligned} x_m(k+i+1) &= A_{md}x_m(k+i) + B_{md}\Delta u_m(k+i) \\ y_m(k+i) &= Cx_m(k+i) \\ x_m(k) &= x_k, \quad \forall \ k, i \geq 0 \end{aligned} \tag{3.42}$$

where x_k is the measured or estimated initial state value from the plant (3.1) and then sampled by T_m at k and $\Delta u_m(k) = u_m(k) - u_m(k-1)$ denotes the control increments or the input rate. These forms of control laws contain automatic integral action in feedback loop and are used for minimum over - shoot and settling times during the online optimization process. Note that integral action rejects constant additive disturbances in the steady state. The computations are realized over receding state prediction N_p and control prediction N_c horizons as $i = 0, 1, ..., N_c, ..., N_p$ at each discrete time instant $k, k+1, k+2, ...$ Assume $N_p = N_c = N$.

The state prediction equation at k is found as

$$\boldsymbol{x}_m(k) = \Theta x_k + \Phi \boldsymbol{\Delta u_m}(k) \tag{3.43}$$

where

$$\begin{aligned}
\boldsymbol{x}_m(k) &= \begin{bmatrix} x_m(k+1) \\ x_m(k+2) \\ \dots \\ x_m(k+N) \end{bmatrix}^T \in \mathbb{R}^{Nn} \\
\boldsymbol{\Delta u_m}(k) &= \begin{bmatrix} \Delta u_m(k) \\ \Delta u_m(k+1) \\ \dots \\ \Delta u_m(k+N-1) \end{bmatrix} \in \mathbb{R}^{Nr}
\end{aligned} \tag{3.44}$$

$$\Theta = \begin{bmatrix} A_{md} & A_{md}^2 & . & . & . & A_{md}^N \end{bmatrix} \in \mathbb{R}^{Nn \times n} \tag{3.45}$$

$$\begin{aligned}
\Phi &= \begin{bmatrix} \Phi_1 & \Phi_2 & . & . & . & \Phi_N \end{bmatrix}^T \\
&= \begin{bmatrix} B_{md} & 0 & \dots & 0 \\ A_{md}B_{md} & B_{md} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{md}^{N-1}B_{md} & A_{md}^{N-2}B_{md} & \dots & B_{md} \end{bmatrix} \in \mathbb{R}^{Nn \times Nr}
\end{aligned} \tag{3.46}$$

In the formulation of UMPC, the hard constraints are used.

The prediction UPM (3.42) is subject to the hard linear state and input rate prediction constraints with T_m

$$\begin{aligned}
&\Delta u_{min} \leq \Delta u_m(k+i) \leq \Delta u_{max} \\
&x_{min} \leq x_m(k+i) \leq x_{max}
\end{aligned} \tag{3.47}$$

We can now express the following UMPC problem with T_m instead of the original

MPC problem with T_p for controlling the plant (3.1) (Kouvaritakis et al., 2016).

Problem 3.1 (UMPC problem for linear dynamical systems)**.**

$$
\begin{aligned}
J_k^*(x_k) = &\min_{\boldsymbol{\Delta u_m}(k)} \; J_k(x_k, \boldsymbol{\Delta u_m}(k)) \\
&= \sum_{i=0}^{N-1} [x_m^T(k+i) q_w x_m(k+i) + \Delta u_m^T(k+i) \\
&\times r_w \Delta u_m(k+i) + x_m^T(k+N) p_N x_m(k+N)] \\
&subject\,to \\
&x_m(k+i+1) = A_{md} x_m(k+i) + B_{md} \Delta u_m(k+i) \\
&\qquad\qquad i = 0, 1, ..., N-1 \; \forall \, k \\
&\Delta u_{min} \leq \Delta u_m(k+i) \leq \Delta u_{max} \\
&\qquad\qquad i = 0, 1, ..., N-1 \; \forall \, k \\
&x_{min} \leq x_m(k+i) \leq x_{max} \\
&\qquad\qquad i = 1, ..., N-1 \; \forall \, k \\
&\Delta u_{min} \leq K(A_{md} + B_{md}K)^i x_m(k+N) \leq \Delta u_{max} \\
&\qquad\qquad i = 0, 1, ..., \epsilon \; \forall \, k \\
&x_{min} \leq (A_{md} + B_{md}K)^i x_m(k+N) \leq x_{max} \\
&\qquad\qquad i = 0, 1, ..., \epsilon \; \forall \, k \\
&x_m(k) = x_k \quad at\,k
\end{aligned}
\tag{3.48}
$$

where $J_k(x_k, \boldsymbol{\Delta u_m}(k))$ is *the objective* or *cost function*, $J_k^*(x_k)$ is *the optimal cost value* or value function, the weighing matrices q_w, (or $q_w = C^T C$), r_w, $x_m(k+N)$ is the terminal state prediction and the terminal weighting matrix p_N are the positive definite. The terminal constraints must satisfy recursively feasibility of the UMPC optimization problem. The UMPC problem can be solved by

- *Batch Approach* after converting to the quadratic programming it,
- recursively *Dynamical Programming Approach*

in discrete - time. We use the batch approach.

This problem for the plant model (3.12) is the original MPC problem. The state feedback gain K is computed over the infinite horizon and unconstrained form of the UMPC algorithm (3.48), it becomes Linear Quadratic Optimal Control Problem or Linear Quadratic Regulator (LQR). The following conditions must be satisfied to be able to ensure closed-loop stability and recursive feasibility:

Assumption 3.1. *A solution exists initially at time $k = 0$ and the UMPC optimization is feasible for all k and the closed - loop system is then stabilized from all initial conditions $x(0)$ under perfect plant model.*

Theorem 3.1 (Asymptotically Stability Theorem of LQR)**.** *If the pair (A_{md}, B_{md}) is stabilizable (controllable), the pair ($A_{md}, q_w^{1/2}$) observable , where is $q_w^{1/2}$ is any matrix and $q_w = (q_w^{1/2})^T q_w^{1/2}$, the state feedback gain $K \in \mathbb{R}^{r\times n}$ is found offline*

$$K = -(B_{md}^T P_\infty B_{md} + R_w)^{-1} B_{md}^T P_\infty A_{md} \tag{3.49}$$

where the weighting matrices

$$Q_w = \begin{bmatrix} q_w & 0 & \dots & 0 \\ 0 & \ddots & & \\ \vdots & & q_w & 0 \\ 0 & \dots & 0 & q_w \end{bmatrix} \in \mathbb{R}^{Nn\times Nn} \tag{3.50}$$

and

$$R_w = \begin{bmatrix} r_w & 0 & \dots & 0 \\ 0 & \ddots & & \\ \vdots & & r_w & 0 \\ 0 & \dots & 0 & r_w \end{bmatrix} \in \mathbb{R}^{Nr\times Nr} \tag{3.51}$$

The unique and positive matrix P_∞ is found offline by the following discrete-time Algebric Ricatti equation (ARE), which is yielded by discrete - time HJB in dynamical programming,

$$P_\infty = A_{md}^T P_\infty A_{md} + Q_w - A_{md}^T P_\infty B_{md} \times (B_{md}^T P_\infty B_{md} + R_w)^{-1} \times B_{md}^T P_\infty A_{md} \tag{3.52}$$

and the all the eigenvalues of $A_{md} + B_{md}K$ lie strictly inside the unit circle.

Theorem 3.2 (Terminal Weighting Matrix p_N)**.** *Along closed - loop trajectories of the plant model (3.23) under the feedback control law $u(k) = Kx(k)$, the asymptotic LQR problem is given by*

$$\begin{aligned} J_k^*(x_i) &= \min_{\boldsymbol{\Delta u_m}(k)} J_k(x_k, \boldsymbol{\Delta u_m}(k)) \\ &= \sum_{i=0}^{\infty} [x_m^T(i) q_w x_m(i) + \Delta u_m^T(i) r_w \Delta u_m(i)] \\ &= x_m^T(0) p_N x_m(0) \end{aligned} \tag{3.53}$$

where the terminal weighting matrix

p_N satisfies the following unique Lyapunov Matrix Equation of the problem (3.53)

$$p_N - (A_{md} + B_{md}K)^T p_N (A_{md} + B_{md}K) = q_w + K^T r_w K \tag{3.54}$$

Definition 25 (Recursive Feasibility)**.** *A control law $u_m(k) = \mu(x_m(k))$ is said recursively feasible for $x(0)$ if $u_m(k) \in \mathcal{U}$ and $x_m(k) \in \mathcal{X}$ along the closed - loop trajectory $x_m(k+1) = A_{md}x_m + B_{md}u_m(k)$ for all $k \in N$*

Definition 26 (Invariant Compact Terminal Constraint Set)**.** *In the UMPC problem, the largest possible terminal constraint set $\mathcal{X}_f$ is defined as*

$$X_f = \begin{cases} x_m(k+N) : \Delta u_{min} \leq K(A_{md} + B_{md}K)^i x_m(k+N) \leq \Delta u_{max}, \\ x_m(k+N) : x_{min} \leq (A_{md} + B_{md}K)^i x_m(k+N) \leq x_{max}, i = 0, 1, ... \end{cases} \tag{3.55}$$

The constraint satisfaction on the largest possible terminal constraint set can be selected checking input constraints over a long enough finite horizon.

Theorem 3.3 (Constraint Checking Horizon)**.** *Let X_{fu} be the set of initial conditions for over the horizon ϵ in the set (3.55)*

$$X_{fu} = (x_m(k+N) : \Delta u_{min} \leq K(A_{md} + B_{md}K)^i x_m(k+N) \leq \Delta u_{max},$$
$$i = 0, 1, \ldots, \epsilon) \quad (3.56)$$

$X_{fu} = X_{Nc} = X_\infty$ *if and only if* $x_m(k) \in X_{Nc+1}$ *whenever* $x_m(k) \in X_{Nc}$.

As a result of the above theorem, the maximum horizon N_c of the terminal constraints for recursively feasibility is found offline by the following linear programs until $\epsilon = N_c$:

$$\begin{aligned}
&\Delta u_{max,j} = \max_x K_j(A_{md} + B_{md}K)^{\epsilon+1} x \\
&s.t. \\
&\Delta u_{min} \leq K(A_{md} + B_{md}K)^i x \leq \Delta u_{max} \\
&i = 0, 1, ..., \epsilon
\end{aligned} \quad (3.57)$$

and

$$\begin{aligned}
&\Delta u_{min,j} = \min_x K_j(A_{md} + B_{md}K)^{\epsilon+1} x \\
&s.t. \\
&\Delta u_{min} \leq K(A_{md} + B_{md}K)^i x \leq \Delta u_{max} \\
&i = 0, 1, ..., \epsilon
\end{aligned} \quad (3.58)$$

where any state $x \in \mathbb{R}^n$ and $j = 0, 1, ..., r$ of the input $u(k) \in \mathbb{R}^r$.

If N is sufficiently long, the closed - loop stability performance of the UMPC increases.

The constraints at k are written in terms of $\boldsymbol{\Delta u_m}$

$$
\begin{bmatrix} \Delta u_{min} \\ \vdots \\ \Delta u_{min} \\ \hdashline x_{min} \\ \vdots \\ x_{min} \\ \hdashline \Delta u_{min} \\ \vdots \\ \Delta u_{min} \\ \hdashline x_{min} \\ \vdots \\ x_{min} \end{bmatrix} \leq \begin{bmatrix} \Delta u_m(k) \\ \vdots \\ \Delta u_m(k+N-1) \\ \hdashline x_m(k+1) \\ \vdots \\ x_m(k+N-1) \\ \hdashline Kx_m(k+N) \\ \vdots \\ K(A+BK)^{\epsilon} x_m(k+N) \\ \hdashline x_m(k+N) \\ \vdots \\ (A+BK)^{\epsilon} x_m(k+N) \end{bmatrix} \leq \begin{bmatrix} \Delta u_{max} \\ \vdots \\ \Delta u_{max} \\ \hdashline x_{max} \\ \vdots \\ x_{max} \\ \hdashline \Delta u_{max} \\ \vdots \\ \Delta u_{max} \\ \hdashline x_{max} \\ \vdots \\ x_{max} \end{bmatrix} \tag{3.59}
$$

The constraint (3.59) is converted in the following compact inequality form

$$
F \boldsymbol{\Delta u_m}(k) \leq G \tag{3.60}
$$

All the matrices in the inequality constraint (3.60) are

$$
F = \begin{bmatrix} F_2 \\ -F_2 \end{bmatrix} \in \mathbb{R}^{2[(N+\epsilon+1)r+(N+\epsilon)n] \times Nr}, \tag{3.61}
$$

$$
G = \begin{bmatrix} F_{max} - F_1 x_k \\ -F_{min} + F_1 x_k \end{bmatrix} \in \mathbb{R}^{2[(N+\epsilon+1)r+(N+\epsilon)n]}, \tag{3.62}
$$

$$F_{min} = \begin{bmatrix} \Delta u_{min} \\ \vdots \\ \Delta u_{min} \\ \hdashline x_{min} \\ \vdots \\ x_{min} \\ \hdashline \Delta u_{min} \\ \vdots \\ \Delta u_{min} \\ \hdashline x_{min} \\ \vdots \\ x_{min} \end{bmatrix}, F_{max} = \begin{bmatrix} \Delta u_{max} \\ \vdots \\ \Delta u_{max} \\ \hdashline x_{max} \\ \vdots \\ x_{max} \\ \hdashline \Delta u_{max} \\ \vdots \\ \Delta u_{max} \\ \hdashline x_{max} \\ \vdots \\ x_{max} \end{bmatrix}, \tag{3.63}$$

$$F_1 = \begin{bmatrix} F_{11} \\ \hdashline F_{12} \\ \hdashline F_{13} \\ \hdashline F_{14} \end{bmatrix}, F_2 = \begin{bmatrix} F_{21} \\ \hdashline F_{22} \\ \hdashline F_{23} \\ \hdashline F_{24} \end{bmatrix}, \tag{3.64}$$

$$F_{11} = \begin{bmatrix} \mathbf{0} \end{bmatrix} \in \mathbb{R}^{Nr \times n}, \tag{3.65}$$

$$F_{12} = \begin{bmatrix} A_{md} & A_{md}^2 & . & . & . & A_{md}^{N-1} \end{bmatrix}^T \in \mathbb{R}^{(N-1)n \times n}, \tag{3.66}$$

$$F_{13} = \begin{bmatrix} KA_{md}^N \\ \vdots \\ K(A_{md} + B_{md}K)^{\epsilon} A_{md}^N \end{bmatrix} \in \mathbb{R}^{(\epsilon+1)r \times n}, \tag{3.67}$$

$$F_{14} = \begin{bmatrix} A_{md}^N \\ \vdots \\ (A_{md} + B_{md}K)^{\epsilon} A_{md}^N \end{bmatrix} \in \mathbb{R}^{(\epsilon+1)n \times n}, \tag{3.68}$$

$$F_{21} = \begin{bmatrix} \boldsymbol{I} \end{bmatrix} \in \mathbb{R}^{Nr \times Nr}, \tag{3.69}$$

$$F_{22} = \begin{bmatrix} \Phi_1 & \Phi_2 & . & . & . & \Phi_{N-1} \end{bmatrix}^T \in \mathbb{R}^{(N-1)n \times Nr}, \tag{3.70}$$

$$F_{23} = \begin{bmatrix} a & \dots & b \\ \vdots & \ddots & \\ c & \dots & d \end{bmatrix} \in \mathbb{R}^{(\epsilon+1)r \times Nr} \tag{3.71}$$

$$a = KA_{md}^{N-1}B, \tag{3.72}$$

$$b = KB_{md}, \tag{3.73}$$

$$c = K(A_{md} + B_{md}K)^{\epsilon} A_{md}^{N-1} B_{md}, \tag{3.74}$$

$$d = K(A_{md} + B_{md}K)^{\epsilon} B_{md}, \tag{3.75}$$

$$F_{24} = \begin{bmatrix} f & \dots & g \\ \vdots & \ddots & \\ h & \dots & i \end{bmatrix} \in \mathbb{R}^{(\epsilon+1)n \times Nr} \tag{3.76}$$

$$f = A_{md}^{N-1}B, \tag{3.77}$$

$$g = B_{md}, \tag{3.78}$$

$$h = (A_{md} + B_{md}K)^{\epsilon} A_{md}^{N-1} B_{md}, \tag{3.79}$$

$$i = (A_{md} + B_{md}K)^{\epsilon} B_{md}.. \tag{3.80}$$

The matrices F and G are computed offline.

The UMPC problem at k is converted to the following QP in terms of $\boldsymbol{\Delta u_m}$

$$\begin{aligned} J_k^*(x_k) &= \min_{\boldsymbol{\Delta u_m}(k)} J_k(x_k, \boldsymbol{\Delta u_m}(k)) \\ &= \boldsymbol{\Delta u_m}^T(k) V \boldsymbol{\Delta u_m}(k) + 2x_k^T Y^T \boldsymbol{\Delta u_m} + x_k^T Z x_k \\ & subject\, to \\ & F\boldsymbol{\Delta u_m}(k) \leq G \end{aligned} \tag{3.81}$$

where $k := kT_m$, $k = 0, 1, ...$

The matrices in the QP (3.81) are given by

$$\begin{aligned} V &= \Phi^T Q_w \Phi + R_w \\ Y &= \Phi^T Q_w \Theta \\ Z &= \Theta^T Q_w \Theta + q_w \end{aligned} \tag{3.82}$$

$$Q_w = \begin{bmatrix} q_w & 0 & \dots & 0 \\ 0 & \ddots & & \\ \vdots & & q_w & 0 \\ 0 & \dots & 0 & p_N \end{bmatrix} \in \mathbb{R}^{Nn \times Nn} \tag{3.83}$$

$$R_w = \begin{bmatrix} r_w & 0 & \dots & 0 \\ 0 & \ddots & & \\ \vdots & & r_w & 0 \\ 0 & \dots & 0 & r_w \end{bmatrix} \in \mathbb{R}^{Nr \times Nr} \tag{3.84}$$

The Lagrangian cost function for the QP (3.81) is

$$\begin{aligned} \boldsymbol{J}(\boldsymbol{\Delta u_m}(k), \lambda) = \boldsymbol{\Delta u_m}^T(k) V \boldsymbol{\Delta u_m}(k) + 2x_k^T Y^T \boldsymbol{\Delta u_m}(k) + x_k^T Z x_k \\ + \lambda^T (F\boldsymbol{\Delta u_m}(k) + G - h) \end{aligned} \tag{3.85}$$

where $\lambda \in R^{Nr}$ and $h \in \mathbb{R}^{2[(N+\epsilon+1)r+(N+\epsilon)n]}$ are the Lagrange multiplier and the slack variable, respectively. The minimum of the Lagrangian cost function (3.85) is found by the following Karush-Kuhn-Tucker (KKT) conditions.

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{\Delta u_m}} = 2V\boldsymbol{\Delta u_m}(k) + 2Yx_k + F^T\lambda = 0, \tag{3.86}$$

$$\frac{\partial \boldsymbol{J}}{\partial \lambda} = F\boldsymbol{\Delta u_m}(k) + G - h = 0, \tag{3.87}$$

$$\lambda \geq 0, \tag{3.88}$$

$$h \geq 0, \tag{3.89}$$

$$h^T\lambda = 0. \tag{3.90}$$

The conditions (3.86-3.87) and (3.88-3.90) are called *feasibility and complementarity conditions*, respectively (Ling et al., 2008). Since the QP (3.81) has the convex cost function and the linear constraints, the computations in the KKT conditions (3.86-3.90) are concluded the global minimum with the feasible initial state. Thus, we can generally write the plant model optimal control law vector $\boldsymbol{\Delta u^*_m}(k)$ as

$$\begin{aligned} \boldsymbol{\Delta u^*_m}(k) &= arg \min_{\boldsymbol{\Delta u_m}(k)} J_k(x_k, \boldsymbol{\Delta u_m}(k)) \\ &\triangleq F_N * x_k \end{aligned} \tag{3.91}$$

over the horizon N at time instant k. $\boldsymbol{\Delta u^*_m}$ is computed online using the improved QP solvers and the nonlinear control laws are computed over the horizon N for any k. If the UMPC problem is written for plant model (3.12), it is the original MPC problem. The original MPC is converted to the following QP with T_p at k

$$\begin{aligned} J^*_k(x_k) = &\min_{\boldsymbol{\Delta u_p}(k)} J_k(x_k, \boldsymbol{\Delta u_p}(k)) \\ &= \boldsymbol{\Delta u_p}^T(k)V\boldsymbol{\Delta u_p}(k) + 2x^T_k(k)Y^T\boldsymbol{\Delta u_p} + x^T_k Z x_k \\ &subject\, to \\ &F\boldsymbol{\Delta u_p}(k) \leq G \end{aligned} \tag{3.92}$$

where $k := kT_p$, $k = 0, 1, ...$

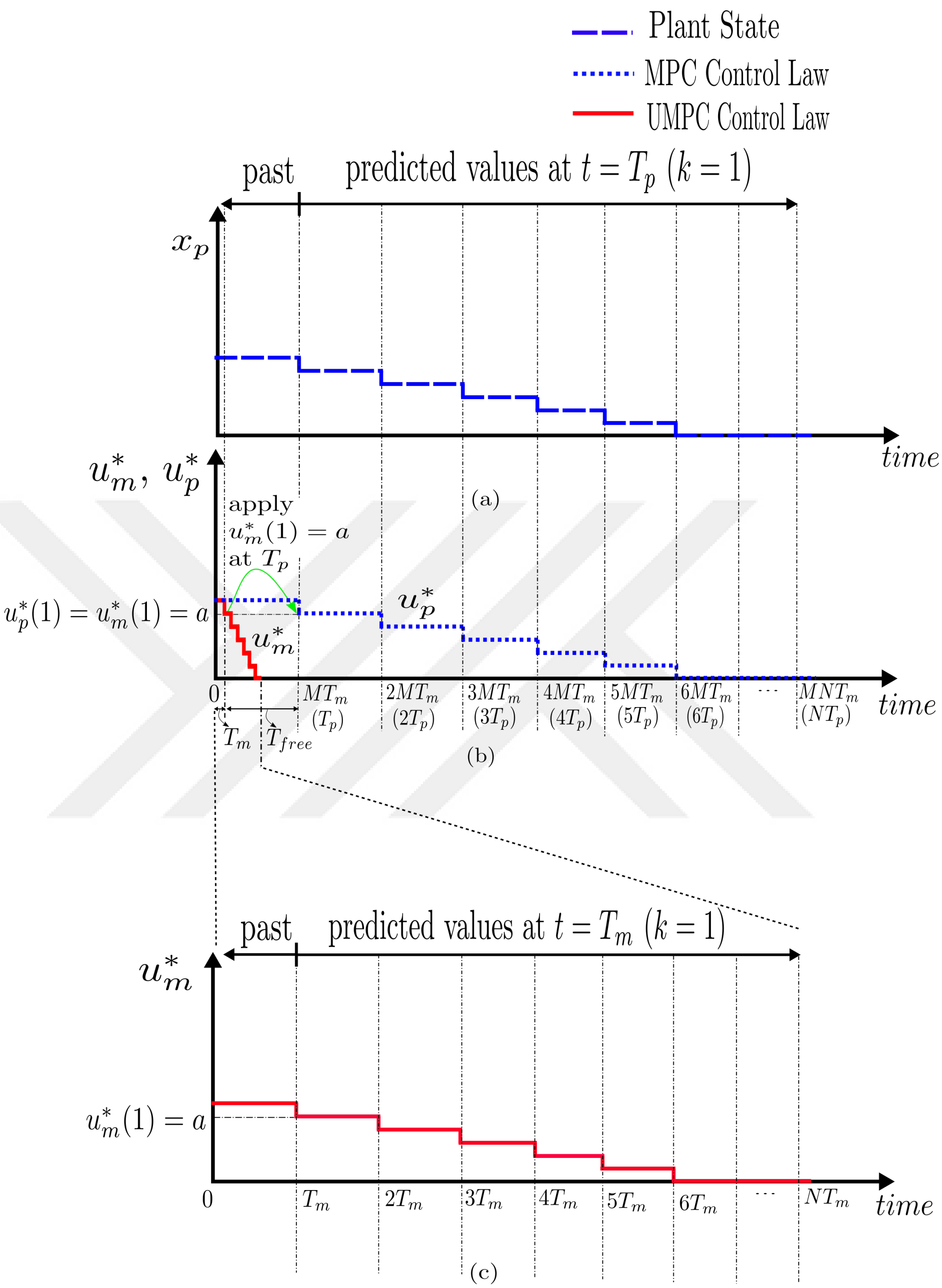

Figure 3.4 The plant control using original MPC and UMPC (a) The predicted plant state response at T_p $(k = 1)$ (b) The predicted input response at T_p $(k = 1)$ using original MPC (c) The fast predicted input response at T_m $(k = 1)$ using UMPC

The plant optimal control law vector $\boldsymbol{\Delta u_p^*}(k)$ is

$$
\begin{aligned}
\boldsymbol{\Delta u_p^*}(k) &= argmin \min_{\boldsymbol{\Delta u_p}(k)} J_k(x_k, \boldsymbol{\Delta u_p}(k)) \\
&\triangleq F_N * x_k
\end{aligned}
\tag{3.93}
$$

over the horizon N at k.

The QPs (3.81) and (3.92) are *time - scaled and sampled version* of each other but instead of $\boldsymbol{\Delta u_p^*}(k)$ at k, which must be implemented to the plant (3.1) or (3.12), $\boldsymbol{\Delta u_m^*}(k)$ is implemented and it is more compatible to microprocessors, which operate fast and parallel.

Thus, we can write the time-shifted relationship between the control laws of MPC and UMPC algorithms for any k as

$$
\boldsymbol{\Delta u_p^*}(kT_p) = \boldsymbol{\Delta u_m^*}(k(T_m + T_{free}))\,\forall\, k \tag{3.94}
$$

and using $\Delta u_m(k) = u_m(k) - u_m(k-1)$

$$
\boldsymbol{u_p^*}(kT_p) = \boldsymbol{u_m^*}(k(T_m + T_{free}))\ \ \forall\, k \tag{3.95}
$$

is computed. M should be selected for the desired T_{free}. The values of $\boldsymbol{\Delta u_m^*}(kT_m)$ are dilated by the free time T_{free} by saving its values in registers of microprocessor and then implemented to the plant at kT_p until next value $(k+1)T_p$ becomes available at each sampling instant k. Thus, $\boldsymbol{\Delta u_m^*}(kT_m)$ can be used instead of $\boldsymbol{\Delta u_p^*}(kT_m)$. Finally, we achieve fast response and free times using UMPC algorithm instead of original MPC algorithm. The predicted responses in original MPC and UMPC at $k = 1$ are illustrated for controlling plant (3.1) or (3.12) in Figure 3.4. This process is repeated until the reference function $r(k)$ on receding horizon N.

Our analysis leads to Algorithm 3.

Algorithm 3 UMPC algorithm

1: Measure the state x_k at kT_m
2: Compute $\boldsymbol{\Delta u_m^*}(kT_m)$ and $\boldsymbol{u_m^*}(kT_m)$ at kT_m
3: If $\boldsymbol{\Delta u_m^*}(kT_m) = 0$ then 'Problem Infeasible' Stop
4: Wait or improve MPC Algorithm in $k(M-1)T_m = kT_{free}$
5: Implement the first element of $\boldsymbol{u_m^*}(kT_m)$ to the plant at kMT_m $(= kT_p)$
6: Wait for the new sampling time $(k+1)MT_m$ $(= (k+1)T_p)$ and then go to step 1

3.2.1 Selecting Time Scaling Factor M

In this work, we suppose that the plant (3.1) must be controlled at times kT_p for $k = 0, 1, ...$ In the control of the dynamical systems, T_p is selected according to the closed - loop bandwidth, settling - time and rise times. In practice, $t_{settling}/100 \leq T_p = T_{rise}/10 \leq t_{settling}/5$ can be selected. $t_{settling}$ and t_{rise} are the settling and rise times of the states of the closed - loop system , respectively, (Bemporad, 2011; Seborg et al., 2010). The states are measured or estimated at each kT_p in receding horizon fashion and required for the computation of the control law in the optimization of the original MPC problem. In this optimization the optimal control law $\boldsymbol{u_p^*}(kT_p)$, the state $\boldsymbol{x_p^*}(kT_p)$ vectors, which optimize the cost function $J_k(x_k, \boldsymbol{\Delta u_p}(k))$, and recursively feasibility checking the constraints on the prediction control $\boldsymbol{u_p}(kT_p)$ and state $\boldsymbol{x_p}(kT_p)$ vectors (3.44) are computed at the computation time periods kT_{pc}. If the computation delays are not available, T_p is ideally assumed to be very much larger than T_{cp}. Finally, the first component of $\boldsymbol{u_p^*}(kT_p)$ is implemented to the plant (3.1) from kT_{pc} until $(k+1)T_p$ becomes available. In the UMPC algorithm, we claim the maximum M (i.e. the minimum T_m instead T_p and the minimum computation time $T_{mp} = T_{cp}/M$) exploiting the speeds of microprocessors used (Elixmann, 2016).

In real time, M depends on the physical constrains (such as speeds of microprocessors, integrated circuits, ADC and DAC, etc.). If a microprocessor or integrated circuit operates at the sampling period $T_{\mu P}$, maximum M can be selected as the following proportion

$$M = T_p/T_{\mu P} \quad \text{for integer} \quad M > 1 \tag{3.96}$$

The sufficient condition for the above proportion is that the sampled values from the plant at T_p must be taken at T_m from the contracted plant, as well. Thus, the constraint for maximum M can be selected as $T_{\mu P}$. Hence, the maximum online optimization sampling period for the UMPC algorithm can be selected as

$$T_m = T_{\mu P} = T_p/M \quad \text{for integer} \quad M > 1 \tag{3.97}$$

Besides, the sampling period of the plant model (3.12) can be selected as

$$T_p^{'} = T_p/L \quad \text{for integer} \quad L \tag{3.98}$$

This equation provides sufficient condition to use the previously computed sequence of the optimal control law u_p^* at current time. In this thesis, the sampling period of the plant model (3.1) is assumed as $T_p^{'} = T_p$ (Dundar & Cihanbegendi, 2019, 2020; Kouvaritakis et al., 2016).

3.3 Numerical Results on Linearized Cessna Aircraft and Discussion

The plant (3.38) or its discrete-time form (3.39) must be controlled at $T_p = 0.2$ $sec.$ using original MPC ($M = 1$) with the plant model (3.39). The MPC and UMPC systems in Figure 3.5 are established to illustrate the application of proposed method on the plant (3.38). Figure 3.5a and 3.5b represent the original MPC and UMPC systems. The UPMs and UMPCs are established and the clocks T_m of them are set for $M = 10, 25, 100$. The control law data computed in the UMPCs are stored for up to each T_{free} in the "UMPCData" block in Figure 3.5b and applied to the plant as open - loop instead the original MPC block in Figure 3.5a. Thus, the control laws of the QPs 3.81 and 3.92 satisfy the condition (3.95).

M depends on the clock speed of MATLAB software installed MATLAB on Intel(R) Core(TM) i5 - 3210M microprocessor with 2.50 GHz and 4 GB RAM. The QPs are run an active set solver MATLAB Model Predictive Control Toolbox (MathWorks, Inc., 2017). The UMPC controllers are attained for $M = 10, 25, 100$. Note that the UMPC controllers for higher M can be simulated using MATLAB.

For the horizon $N = 10$, the matrices $q_w = I$, $r_w = 10$, the pitch reference value 0 $rad.$, the altitude reference value 5000 $m.$ and $M = 1, 10, 25, 100$, the responses and optimal control laws of MPC and UMPC controllers are illustrated in Figure 3.6.

It proves that UMPC controllers settle faster than original MPC to the reference values and there exist the free sampling periods $T_{free} = 0.18, 0.192$ and 0.198 $sec.$ between the original MPC and UMPC controllers and when an improvement for the control is not made, the free times mean energy efficiency for UMPC.

The performances of the controllers are compared in Table 3.1 and Table 3.2. For each M and 170 iterations Table 3.1 shows the rise and settling time responses of the controllers and Table 3.2 shows the free times and the energy values of the control signals. Energy value E of a signal $u(t)$ is computed by the following equation

$$E = \int_{-\infty}^{\infty} \mid u(t) \mid^2 dt \tag{3.99}$$

In the tables, adapting MPC algorithm (i.e. UMPC) by the selected scaling factors depending on the speed of the microprocessor, the controller energy and settling time decreases dramatically. As illustrated in Figure 3.6, the first value of the computed optimal control law vector at kT_m for each UMPC controller is dilated by kT_{free} by saving its first value. Thus, the desired fast - response UMPC can be used instead of the original MPC.

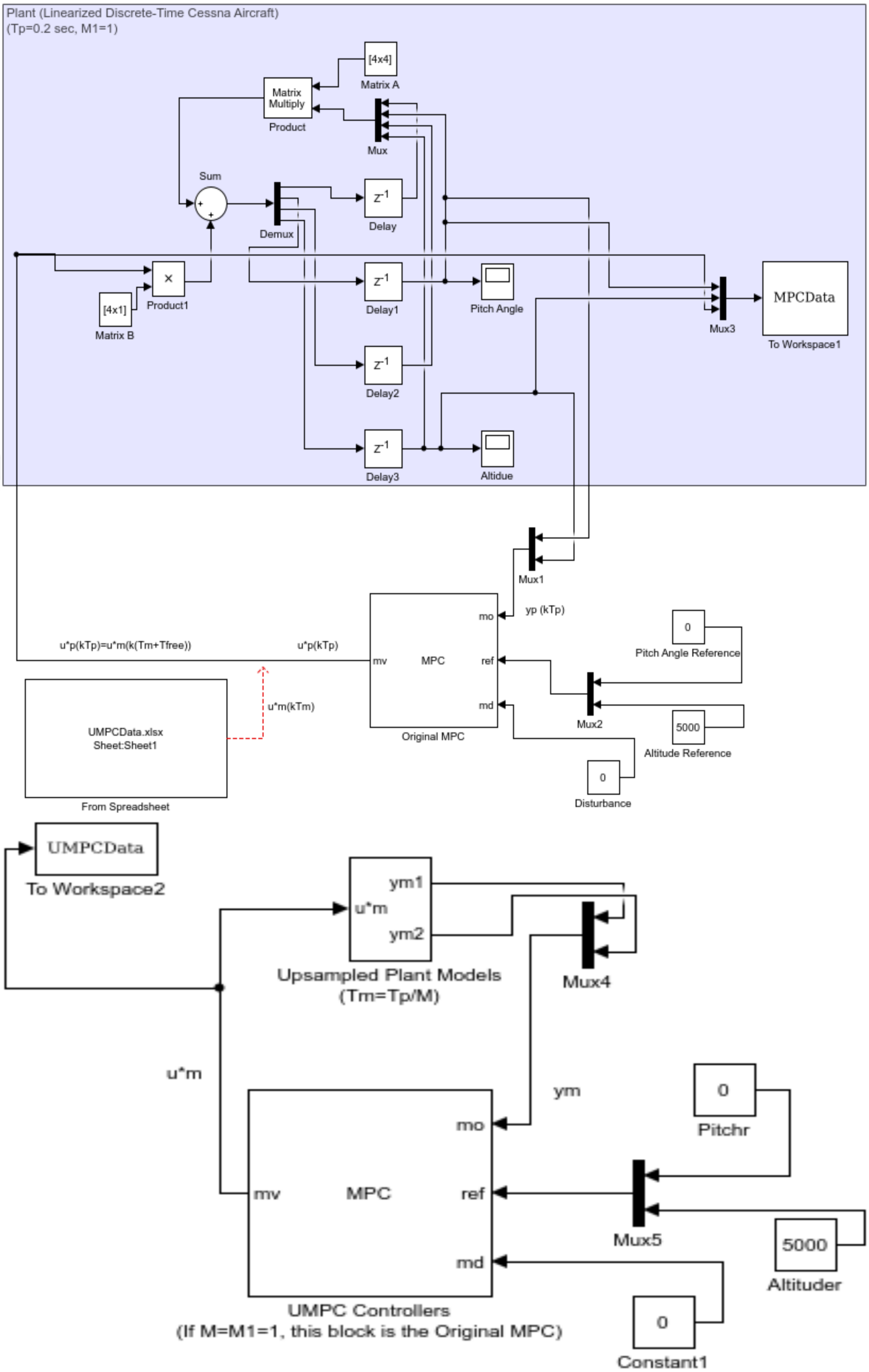

Figure 3.5 The establishment on Simulink for the control of Cessna aircraft (a) Original MPC system ($M = 1$) (b) UMPC systems ($M = 10, 25, 100$)

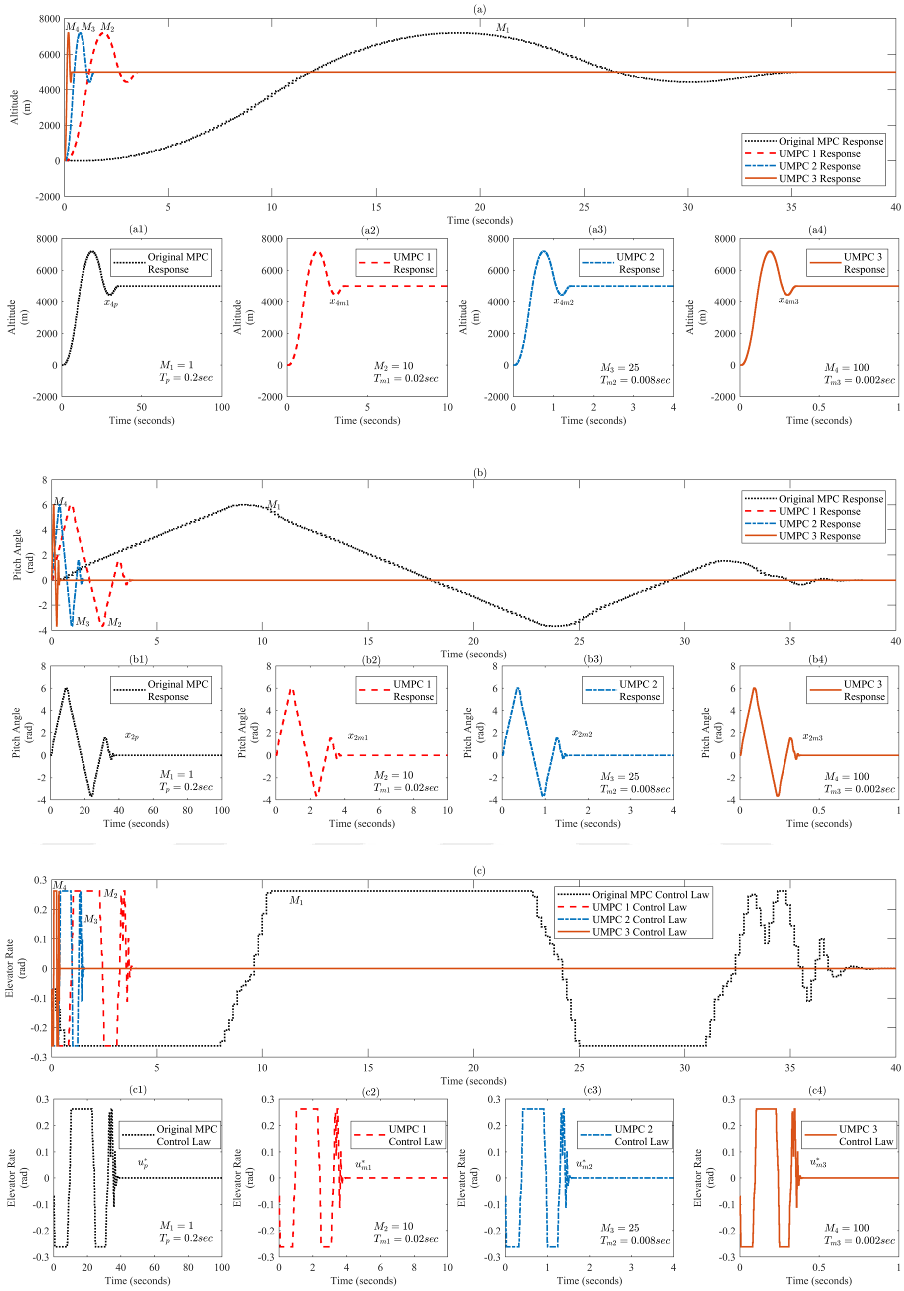

Figure 3.6 The Control laws and responses of original MPC and UMPC for $M = 1, 10, 25, 100$ (a) The altitude responses of MPC and UMPC (b) The pitch responses of MPC and UMPC (c) The control laws of MPC and UMPC

Since the UMPC is compatible with the microprocessor and the origin of speed of the UMPC is microprocessor, slow operation issue of MPC algorithms is solved for different time scaling factors by this method without reducing computation load of online optimization of used MPC algorithms. We used the standard MPC problem as the original MPC and compared it with the UMPC. The method can be applied to

Table 3.1 Original MPC and UMPC responses

Controllers	Scaling Factor M	Number of Iterations	Pitch / Altitude Rise Time (sec)	Pitch / Altitude Settling Time (sec)
Original MPC	1	170	1.810 / 6.813	34
UMPC 1	10	170	0.1810 / 0.6813	3.4
UMPC 2	25	170	0.0724 / 0.2722	1.36
UMPC 3	100	170	0.0181 / 0.0681	0.34

Table 3.2 Original MPC and UMPC control laws

Controllers	Free Time Sampling Period (sec)	Number of Iterations	Total Free Time until Settling Time (sec)	Control Energy
Original MPC	0	170	0	1.9502
UMPC 1	0.18	170	30.6	0.1950
UMPC 2	0.192	170	32.64	0.0780
UMPC 3	0.198	170	33.66	0.0195

the methods in the references and nonlinear dynamical systems, as well. For example the method can be applied to Time Scale Seperation method in the reference (Zhang et al., 2013). For study a high-scaled plant is separated into slow and fast dynamics. These dynamics are converted to slow and fast UPMs by scaling and sampling using a time scaling factor. Then, UMPCs are obtained based on UPMs. The UMPCs settle to desired output faster than the original slow and fast MPCs and have free times and lower control energies.

In real - time or practical applications, some modifications on plant model, constraints, optimization, receding horizon strategy of MPC algorithms must be done to be able to implement to fast plants of the theoretical MPC methods. These modifications can be classified under the title *Real-Time, Hard and Soft Constrained MPC Methods*. Although MPC is theoretically sound established and there are some methods with significant limits for fast plants, which the methods are based on direct optimization methods, it is more convenient method in slow systems for real - time applications under the online computing load given detailed in Chapter 1.

Constraints are evaluated as hard and soft. Since input and input rate constraints have physical limitations (such as system requirements and safety), they are considered as hard constraints.

$$
\begin{aligned}
&\Delta u_{min} \leq \Delta u(k) \leq \Delta u_{max} \\
&u_{min} \leq u(k) \leq u_{max}
\end{aligned}
\tag{3.100}
$$

Since the state and output constraints may be violated because of disturbances and / or plant - plant model mismatches, they are considered as hard or soft considered. Soft constraints are defined by

$$
\begin{aligned}
&x_{min} + s_x \leq x(k) \leq x_{max} + s_x \\
&y_{min} + s_y \leq y(k) \leq y_{max} + s_y
\end{aligned}
\tag{3.101}
$$

where s_x and s_y are the slack variables (Wang, 2008). The constraints are handled in the MPC computations and increase the computation dimension. Therefore they causes computation burden and in addition while soft constraints satisfy feasible solutions compensating mismatches between plant and plant model, to choose slack variable is extra computation effort in optimization process.

MPC algorithms for LTI dynamical systems compute the control laws online and offline by minimizing convex or quadratic optimization problem in direct optimization methods. In the optimization problem modified on these methods, the optimal solutions can be reached using Newton - based numerical methods. The fast methods for the solution of MPC problems used in the literature are briefly described below. Refer to (Borrelli et al., 2015; TøNdel et al., 2003; Rao et al., 1998; Bemporad et al., 2002; Fletcher, 2013; Cai et al., 2014; Borrelli et al., 2010; Wang & Boyd, 2010; Yildirim & Wright, 2002; Koehler et al., 2017; Morari, 2013; Zeilinger, 2011; Liu et al., 2018; Shahzad et al., 2010), (Zhang et al., 2013) and (Zannon, 2015; Paiva, 2014; Rawlings et al., 2019; Primbs et al., 1999; Chen, 2016) for the details

3.3.1 Hard and Soft Constrained Explicit MPC Methods

In these methods, the online computational burden of the original MPC strategy is moved to offline computations. The optimal control law vectors are written as the pre-defined piecewise affine function of the feasible state $x(k)$ over a polyhedral partition $\mathscr{P}$

$$u^*(k) \triangleq a_i x(k) + b_i \;\; x(k) \in \mathscr{P}_i, \forall \mathscr{P}_i \in \mathscr{P} \tag{3.102}$$

This equation is numerically obtained as pre-computed and stored as a look - up table computed by *multiparametric programming methods*. Since the computational complexity of the method increases with the problem size (plant, horizon and constraints), there are storage space requirements. It is convenient and fast for small - sized MPC problems due to computational complexity and storage space requirements.

3.3.2 Hard and Soft Constrained Implicit MPC Methods

The direct implementation of the original MPC strategy for high dimensional systems is in this category. The optimization is done by online computations and computational burden is high - level. There are *warm - start interior point* and *active set methods* for controlling fast plants. Warm start is used the previous solution for the next iteration and selected a feasible initial point where it is close to the optimum for decreasing computation time. The structure of MPC formulations can be reduced to the sparse formulations of the optimal control problems.

In the literature, the implicit MPC methods are still the research topics.

3.3.3 Model Reduction Techniques on Plant Model of MPC

Higher - order plant models are reduced into lower - order sub plant models or divided into fast and slow sub plant models. This approach is also called *Time Scale Separation*. The MPC algorithms are constructed on them and decreases the computation load (see Appendix 1).

Since the UMPC is a general approach to the above methods, it can be more effectively implemented on the methods for controlling the fast systems

3.4 Conclusion

In this thesis, we introduced the UMPC method for the LTI dynamical systems. In the method, the original MPC strategy is contracted by using a time scaling factor based on the speed of the microprocessor in time domain. The original MPC is transformed to the UMPC. Therefore, the QPs of the original MPC and UMPC is the same except the online optimization sampling periods. The results prove that UMPC is settled to the reference values faster than the original MPC and provides lower control energy. In addition, since there exists free time periods between the original MPC and UMPC online optimization sampling periods, the first values of computed control law vectors in the UMPC are dilated to be able to equate it to the optimization sampling periods of the original MPC by the free time periods and then are applied to the plant at original optimization sampling periods.

In addition, in free times, the UMPC algorithm can be improved by

- compensating modelling errors and disturbances by online system identification,
- increasing control capabilities using different control scenarios (such as optimization computations on varying horizon,
- incorporating with other control strategies (such as adaptive control and artificial intelligence algorithms).

CHAPTER 4

UPSAMPLED MODEL PREDICTIVE CONTROL FOR NONLINEAR DYNAMICAL SYSTEMS

In control theory, feedback linearization method for controlling of a class of nonlinear dynamical systems (plants) is a classical approach. The aim is to transform a class of nonlinear systems into an equivalent linear dynamical system by state feedback. The nonlinear terms of nonlinear system are cancelled by state feedback and are moved into the control law. Thus, linear control theory can be applied for the linearized systems obtained by this way. The feedback linarization technique is demonstrated in Figure 4.1. Many control algorithms can be applied on the feedback linearization method. One of the most important control algorithms is Model Predictive Control (MPC). MPC applications which can be applied successfully on linearized plant model are available in the literature (Hafez et al., 2015; Schnelle & Eberhard, 2015; Orchi et al., 2018).

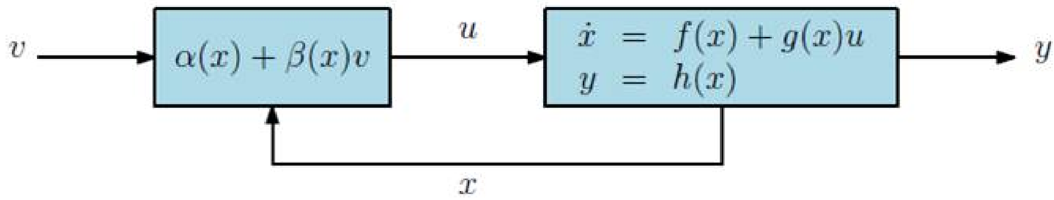

Figure 4.1 Input - output feedback linerization

In this chapter, we present the nonlinear version of the approach proposed in Chapter 3. The dynamics of the linear model (plant model) obtained by feedback linearization method of a nonlinear dynamic system are sped up and sampled using an integer time scaling factor M. As a result of these modifications, the linearized contracted and upsampled plant model for $M > 1$ are achieved. When $M = 1$, the plant of the nonlinear dynamical system and the plant model used in the nominal MPC are obtained. The UMPC algorithm, which is the sped up version of the original MPC, is constructed by the linearized upsampled plant model and the linearized plant of the nonlinear system is controlled.

The MPC strategy is rendered fast-response with this modification in the time domain, fast-prediction and low-control energy at each optimization period and can be applied to fast dynamical systems similar to Chapter 3. The success of the method is demonstrated on the PVTOL dynamics.

This chapter is organized as follows: The Linearized UPM is obtained from nonlinear plant model in Section 4.1, the relationship between the control laws computed in the original MPC and UMPC algorithms is formulated in Section 4.2, the fast response, free time and lower control energy contributions of UMPC are illustrated on PVTOL aircraft dynamics and compared to the original MPC in Section 4.3, and the conclusion follows in Section 4.4.

4.1 Linearized Time - Scaled and Upsampled Solutions of Nonlinear Dynamical Systems

Consider the nonlinear dynamical system (plant model) (Sastry, 1999)

$$\begin{aligned} \frac{dx(t)}{dt} &= M[f(x(t)) + \sum_{i=1}^{r}(g_i(x(t))u_i(Mt))], \\ y_1(t) &= h_1(x(t)), \\ &. \\ &. \\ &. \\ y_r(t) &= h_r(x(t)), \\ x(0) &= x_0, \quad \forall \ t \geq 0 \end{aligned} \tag{4.1}$$

where state variable $x(t) \in \mathbb{R}^n$, input variable $u(t) \in \mathbb{R}^r$, output variable $y(t) \in \mathbb{R}^r$, f, g_i are smooth nonlinear vector fields, h_j is smooth nonlinear scalar function, M is the integer time scaling factor for the contracted solution of plant model and continuous-time variable $t \in \mathbb{R}^+$. The parameters according to the condition of M are shown in Table 4.1.

In case $M = 1$ for the plant model (4.1), we will define the original plant model, and for other integer values of M, we will define it as contracted plant models. Assume

Table 4.1 Parameters

Plant Models	**Input, State and Output Variables**	**Integer Time Scaling Factor M**	**Controller Type**
Original Plant Model	u_p, x_p, y_p	1	Original MPC
Contracted Plant Model	u_m, x_m, y_m	otherwise	Upsampled MPC

that the plant models (4.1) for different M are minimal and have unique solution. The undriven form of systems have the same equilibrium point x_{eq}.

Since we want completely to scale the solution of the plant model by M in time domain, the contracted solution for $u(t)$ must be modified as $u(Mt)$.

We want to obtain input - output linear plant model from the plant model. For this aim the derivatives of $y_j(t)$ with respect to time are taken until at least one of the inputs appears $u_i(Mt)$. The following equation is obtained by the first derivative of $y_j(t)$.

$$\frac{dy_j}{dt} = \dot{y}_j(t) = M[L_f h_j(x(t)) + \sum_{i=1}^{r} L_{g_i} h_j(x(t)) u_i(Mt)] \tag{4.2}$$

where
$L_f h_j(x(t)) = \frac{\partial h_i(x(t))}{\partial x(t)} f(x(t))$ and $L_{g_i} h_j(x(t)) = \frac{\partial h_j(x(t))}{\partial x(t)} g_i(x(t))$ are the Lie derivatives of h_j with respect to f and g_i.

If $L_g h(x(t))(x(t)) \equiv 0$, the following equation is obtained by the second derivative of $y_j(t)$.

$$\ddot{y}_j(t) = M^2[L_f^2 h_j(x(t)) + \sum_{i=1}^{r} L_{g_i} L_f h_j(x(t) u_i(Mt)] \tag{4.3}$$

where $L_f^2 h_i(x(t) = L_f(L_f h_i(x(t)) = \frac{\partial L_f h_i(x(t))}{\partial x(t)} f(x(t))$ and $L_{g_j} L_f h_i(x(t)) = L_{g_j}(L_f h(x(t))) = \frac{\partial L_{g_j} h_j(x(t))}{\partial x(t)} g_i(x(t))$.

Generally, if $L_g L_f^j h(x(t)) \equiv 0$ for $j = 0, 1, ..., \gamma_i - 2$ for the smallest integer γ_i, the following equation is obtained by the γ_i-th derivative of $y_j(t)$.

$$y_j^{\gamma_i}(t) = M^{\gamma_i}[L_f^{\gamma_i} h_j(x(t)) + \sum_{i=1}^{r} L_{g_i} L_f^{\gamma_i - 1} h_j(x(t)) u_i(Mt)] \tag{4.4}$$

with at least one of the $L_{g_i} L_f^{\gamma_i - 1} h_j(x(t)) \neq 0$ for some x. Thus, at least one of the inputs appears in $y_j^{\gamma_i}(t)$ and the $r \times r$ matrix $T(x)$ is defined as

$$T(x) = \begin{bmatrix} M^{\gamma_1} L_{g_1} L_f^{\gamma_1 - 1} h_1 & \dots & M^{\gamma_1} L_{g_r} L_f^{\gamma_1 - 1} h_1 \\ \vdots & \ddots & \vdots \\ M^{\gamma_r} L_{g_1} L_f^{\gamma_r - 1} h_r & \dots & M^{\gamma_r} L_{g_r} L_f^{\gamma_r - 1} h_r \end{bmatrix} \tag{4.5}$$

$T(x)$ is called decoupling matrix.

Definition 27 (Vector Relative Degree)**.** *The plant model (4.1) has vector relative degree $\gamma_1, \gamma_2, ...\gamma_r$ at x_{eq} if*

$$L_{g_i} L_f^k h_i(x(t)) \equiv 0, \quad 0 \leq k \leq \gamma_i - 2 \quad for\ i = 1, ..., r \tag{4.6}$$

and the $T(x_{eq})$ is nonsingular.

If the equation (4.4) is rewritten in the matrix form, the following equation is obtained

$$\begin{bmatrix} y_j^{\gamma_1}(t) \\ \vdots \\ y_j^{\gamma_r}(t) \end{bmatrix} = \begin{bmatrix} M^{\gamma_1} L_f^{\gamma_1} h_1 \\ \vdots \\ M^{\gamma_r} L_f^{\gamma_r} h_r \end{bmatrix} + T(x) \begin{bmatrix} u_r(Mt) \\ \vdots \\ u_r(Mt) \end{bmatrix} \tag{4.7}$$

Since $T(x)$ is nonsingular, the static state feedback control law

$$u_p(x(t)) = T^{-1}(x) \begin{bmatrix} M^{\gamma_1} L_f^{\gamma_1} h_1 \\ \vdots \\ M^{\gamma_r} L_f^{\gamma_r} h_r \end{bmatrix} + T^{-1}(x) \upsilon(Mt) \tag{4.8}$$

obtains the γ-th order closed loop linearized plant model of the plant model (4.1) from the new input $\upsilon_i(Mt)$ to the output $y_j(t)$

$$\begin{bmatrix} y_1^{\gamma_1}(t) \\ \vdots \\ y_r^{\gamma_r}(t) \end{bmatrix} = \begin{bmatrix} M^{\gamma_1}\upsilon_1(Mt) \\ \vdots \\ M^{\gamma_r}\upsilon_r(Mt) \end{bmatrix} \tag{4.9}$$

$\gamma = \gamma_1 + ... + \gamma_r$ is defined *Total Strict Relative Degree* of the plant model (4.1). If $\gamma = n$, there does not exist *Internal Dynamics* or *Zero Dynamics* (see Appendix 2) and Linear System Theory can be applied. In this work, $\gamma = n$ is assumed. If $T(x)$ is singular and $\gamma < n$, *Dynamical Extension Algorithm* is applied and the augmented linear plant models are obtained. In Section 4.1, this algorithm is applied on PVTOL dynamics (Sastry, 1999).

The states and state equation of the linearized plant model are obtained by the following coordinate transformation and the definitions

$$z(t) = \phi(x(t)) = \begin{bmatrix} z_1(t) & z_2(t) & . & . & . & z_n(t) \end{bmatrix}^T \tag{4.10}$$

$$\begin{aligned}
& y_1(t) \triangleq z_1(t) \\
& \dot{y}_1(t) = \dot{z}_1(t) \triangleq Mz_2(t) \\
& \ddot{y}_1(t) = M\dot{z}_2(t) \triangleq M^2 z_3(t) \\
& \vdots \\
& y_1^{\gamma_1}(t) = M^{\gamma_1 - 1}\dot{z}_{\gamma_1}(t) \triangleq M^{\gamma_1}\upsilon_1(Mt) \\
& y_2(t) \triangleq z_{\gamma_1+1}(t) \\
& \vdots \\
& y_2^{\gamma_2}(t) = M^{\gamma_2 - 1}\dot{z}_{\gamma_1+\gamma_2}(t) \triangleq M^{\gamma_2}\upsilon_2(Mt) \\
& \vdots \\
& y_r(t) \triangleq z_{\gamma_1+\gamma_2+...+\gamma_{r-1}+1}(t) \\
& \vdots \\
& y_r^{\gamma_r}(t) = M^{\gamma_r - 1}\dot{z}_n(t) \triangleq M^{\gamma_r}\upsilon_r(Mt)
\end{aligned} \tag{4.11}$$

where $z \in \mathbb{R}^n$ and $\phi(.) : x \mapsto \phi(x) = z$ is a diffeomorphism map. Thus, the $\gamma = n$-th order compact linearized plant model is obtained as

$$\begin{aligned} \dot{z}(t) &= M(Az(t) + B\upsilon(Mt)) \\ y_1(t) &= z_1(t) \\ z(0) &= z_0 \end{aligned} \tag{4.12}$$

where

$A = \begin{bmatrix} 0 & 1 & 0 & . & . & . & 0 \\ 0 & 0 & 1 & . & . & . & 0 \\ . & . & . & . & & & . \\ . & . & . & & . & & . \\ . & . & . & & & . & . \\ 0 & 0 & 0 & . & . & . & 1 \\ 0 & 0 & 0 & . & . & . & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ . \\ 1 \end{bmatrix}$ are in controllable canonical form.

The state space (4.12) has the transfer function $(M/s)^n$ with n-integrator. The solution similar to the equation (3.10) for the linearized plant model (4.12) is

$$\begin{aligned} z(Mt) &= z_0 e^{MAt} + M e^{MAt} \int_0^t e^{-MA\tau} B\upsilon(M\tau) d\tau \\ &= \begin{cases} z(t) \triangleq z_p(t), & \text{if } M = 1 \\ z(Mt) \triangleq z_m(t), & \text{if otherwise } M \end{cases} \end{aligned} \tag{4.13}$$

where $z_p(t)$ is the original plant model solution with $\upsilon(t)$ and $z_m(t)$ is called *the Contracted Plant Model Solution* with $\upsilon(Mt)$. $z_m(t)$ has faster response by M times than the original plant model $z_p(t)$ in time domain.

The discrete-time form of the plant model (4.12) is given by

$$\begin{aligned} z(k+1) &= A_d z(k) + B_d \upsilon(k) \\ y(k) &= C_d z(k) \\ z(0) &= z_0, \quad \forall \ k \geq 0 \end{aligned} \tag{4.14}$$

Assume the original plant model with $M = 1$ is sampled and controlled at sampling period $k := kT_p$, which is the optimization sampling period of the original MPC algorithm with the discrete-time form $\upsilon_p(T_pk) := \upsilon_p(k)$ of $\upsilon(t)$. If the contracted plant models with $M > 1$ are sampled and controlled at $k := kT_m = kT_p/M$, which is the optimization sampling period of the Upsampled MPC algorithm with the discrete-time form $\upsilon_p(T_mk) := \upsilon_p(k)$ of $\upsilon(Mt)$, we call *Linearized Contracted and Discrete-Time of the Original Discrete-Time Plant Model or Linearized Upsampled Plant Model.*

In this case, there exists the following time-shifting relationship between the discrete - time solutions of the plant models

$$z_p(kT_p) = z_m(k(T_m + T_{free})) \;\; \forall\, k \tag{4.15}$$

where

$$\begin{aligned} T_{free} &= T_p - T_m \\ &= T_m(M-1) \end{aligned} \tag{4.16}$$

The equation (4.15) can be expressed such that when we obtain the linearized UPMs for different $M > 1$ by means of parallel and fast microprocessor, we need to store their values for the free sampling periods T_{free} at each time instant k to synchronize these models to the linearized original plant model.

4.2 Control Laws of Original MPC and UMPC

Assume that the original MPC algorithm based on the original plant model with $M = 1$ performs the control of the real plant at each optimization sampling period T_p, which is chosen according to the bandwidth and settling time of closed - loop plant model. Instead, we want to perform the control of the real plant using the same MPC algorithm based on the UPMs with $M > 1$ each sampling period T_m. We assume that there are no other modelling errors between the real plant and plant models. The standard MPC formulation is used for the original and contracted plant models.

The prediction plant model of the state space (4.14) for any M is

$$
\begin{aligned}
z(k+i+1) &= A_d z(k+i) + B_d \upsilon(k+i) \\
y(k+i) &= C_d z(k+i) \\
z(k) &= z_k, \quad \forall \;\; k, i \geq 0
\end{aligned}
\tag{4.17}
$$

where z_k is the measured or estimated initial state value at k.

The predicted state and control input variables are

$$
\begin{aligned}
\boldsymbol{z}(k) &= \begin{bmatrix} z(k) & z(k+1) & \dots & z(k+N) \end{bmatrix}^T \\
\boldsymbol{\upsilon}(k) &= \begin{bmatrix} \upsilon(k) & \upsilon(k+1) & \dots & \upsilon(k+N-1) \end{bmatrix}^T
\end{aligned}
\tag{4.18}
$$

over each receding prediction horizon $k, k+1, ...N$.

The prediction UPM (4.17) is subject to the linear state and input prediction constraints

$$
\begin{aligned}
\upsilon_{min} \leq \upsilon(k+i) \leq \upsilon_{max} \\
z_{min} \leq z(k+i) \leq z_{max}
\end{aligned}
\tag{4.19}
$$

We can now write the following UMPC problem with T_m instead of the original MPC problem with T_p

Problem 4.1 (UMPC problem for feedback - linearized dynamical systems)**.**

$$
\begin{aligned}
J_k^*(z_k) &= \min_{\boldsymbol{\upsilon}(k)} \; J_k(z_k, \boldsymbol{\upsilon}(k)) \\
&= \sum_{i=0}^{N-1} \parallel z(k+i) \parallel_{q_w}^2 + \parallel \upsilon(k+i) \parallel_{r_w}^2 + \parallel z(k+N) \parallel_{p_N}^2 \\
& subject\, to \\
& z(k+i+1) = A_d z(k+i) + B_d \upsilon(k+i) \\
& \qquad\qquad i = 0, 1, ..., N-1 \; \forall \, k \\
& \upsilon_{min} \leq \upsilon_m(k+i) \leq \upsilon_{max} \\
& \qquad\qquad i = 0, 1, ..., N-1 \; \forall \, k \\
& z_{min} \leq z(k+i) \leq z_{max} \\
& \qquad\qquad i = 1, ..., N-1 \; \forall \, k \\
& z(k+N) \in \mathscr{Z} \\
& z(k) = z_k \quad at\, k
\end{aligned}
\tag{4.20}
$$

where $J_k(z_k, \boldsymbol{\upsilon}(k))$ is the cost or objective function, $J_k^*(z_k)$ is the optimal cost value, the weighted matrices q_w, (or $q_w = C_d^T C_d$), r_w and p_N are the positive definite, $z(k+N)$ is the terminal state prediction and $\mathscr{Z}$ is the invariant compact convex terminal set. We call this MPC problem as *Upsampled MPC (UMPC)* problem.

The UMPC problem is converted to the following QP in terms of $\boldsymbol{\upsilon}$ for different M values.

$$
\begin{aligned}
J_k^*(z_k) &= \min_{\boldsymbol{\upsilon}(k)} \; J_k(z_k, \boldsymbol{\upsilon}(k)) \\
&= \boldsymbol{\upsilon}^T(k) V \boldsymbol{\upsilon}(k) + 2 z_k^T Y^T \boldsymbol{\upsilon} + z_k^T Z z_k \\
& subject\, to \\
& F \boldsymbol{\upsilon}(k) \leq G
\end{aligned}
\tag{4.21}
$$

where the matrices V,Y, Z, F and G are computed offline over moving N and at each $k := kT_m$, $k = 0, 1, ...$

The Lagrangian cost function for the QP (4.21) is

$$\boldsymbol{J}(\boldsymbol{v}(k), \lambda) = \boldsymbol{v}^T(k)V\boldsymbol{v}(k) + 2x_k^T Y^T \boldsymbol{v}(k) + z_k^T Z z_k + \lambda^T(F\boldsymbol{v}(k) + G - h) \quad (4.22)$$

where λ and h are the Lagrange multiplier and the slack variable, respectively. Since the plant models are linear and has linear inequalities, the solution is the global minimum. The minimum Lagrangian cost function (4.22) is determined by the following KKT conditions.

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{v}} = 2V\boldsymbol{v}(k) + 2Yz_k + F^T\lambda = 0, \quad (4.23)$$

$$\frac{\partial \boldsymbol{J}}{\partial \lambda} = F\boldsymbol{v}(k) + G - h = 0, \quad (4.24)$$

$$\lambda \geq 0, \quad (4.25)$$

$$h \geq 0, \quad (4.26)$$

$$h^T\lambda = 0. \quad (4.27)$$

The conditions (4.23-4.24) and (4.25-4.27) are the feasibility and complementary conditions, respectively. The QP (4.21) has the convex cost function and the linear constraints. Therefore, the solution is a global minimum in the feasible initial conditions.

Since the UMPC is the time-shifted version of the original MPC, the computed optimal control law vectors are also the time-shifted version of each other. This relationship is given below

$$\begin{aligned} \boldsymbol{v}^*(k) &= arg \min_{\boldsymbol{v}(k)} J_k(z_k, \boldsymbol{v}(k)) \\ &= \begin{cases} \boldsymbol{v}_{\boldsymbol{p}}^*(k) \triangleq F_N * z_k, & \text{if } M = 1 \\ \boldsymbol{v}_{\boldsymbol{m}}^*(k) \triangleq F_N * z_k, & \text{if otherwise } M \end{cases} \end{aligned} \quad (4.28)$$

over the horizon N at time instant k. $\boldsymbol{v}^*(k)$ is computed online using the improved QP solvers and the nonlinear control laws are computed over the time horizon N for any k.

Thus, we can write the time-shifting relationship between the control laws computed in MPC and UMPC algorithms for any k as

$$\boldsymbol{\upsilon_p^*}(kT_p) = \boldsymbol{\upsilon_m^*}(k(T_m + T_{free}))\ \forall\, k \tag{4.29}$$

The attractive side of this equation is that the original MPC algorithm based on the feedback linearized plant model with $M = 1$ is modified to adapt the algorithm to the speed of the processor to completely eliminate the computational load due to online optimization. In the equation, since the UMPC based on the feedback linearized with $M > 1$ takes place faster than the original MPC, the free sampling periods T_{free} become. Fast computed values should be stored in the free time periods. In this case, $\boldsymbol{\upsilon_m^*}$ can be applied to the real plant instead of $\boldsymbol{\upsilon_p^*}$.

Notice the source of M is the speed of microprocessor used. If the microprocessor is capable of sampling and online optimization at desired sampling periods T_m, larger M values can be selected as long as the original plant model values obtained by sampling with T_p from the real plant can also be sampled for $M > 1$ or T_m.

4.2.1 Original MPC and UMPC Applications on Van der POL Oscillator Dynamics

Consider the controlled simple Van der Pol oscillator shown in Figure 4.2. Here the inductance L and the capacitance C are linear and the resistance R is a negative nonlinear resistor that has the cubic characteristic $i_R(\nu_C(t)) = \beta\nu_C(t)(\nu_C^2(t) - 1)$ with $\beta > 0$. Using Kirchoff's laws, the equations are attained as

$$\begin{aligned}
&L\dot{i}_L(t) = \nu_C(t) \\
&C\dot{\nu}_C(t) = -i_L(t) - i_R(\nu_C(t)) + \frac{\nu_C(t) - u(t)}{r} \\
&i_L(0) = i_{L0} \\
&\nu_C(0) = \nu_{C0}
\end{aligned} \tag{4.30}$$

The inductor current $i_L(t)$ and capacitor voltage $\nu_C(t)$ is chosen as the state variables $i_L(t) \triangleq x_1(t)$, $\nu_C(t) \triangleq x_2(t)$. The output $y(t) \triangleq x_1(t)$ and $i_R(x_2(t)) = \beta x_2(t)(x_2^2(t) - 1)$. The output inductance current $y(t) = x_1(t)$ will be controlled by the input / control voltage $u(t)$.

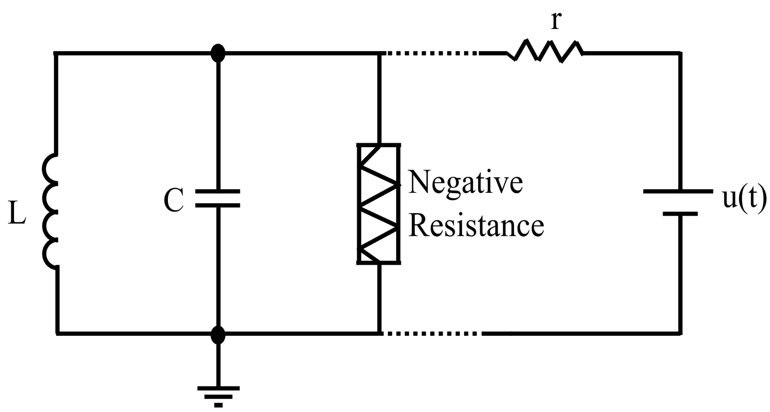

Figure 4.2 Van der Pol oscillator (Sastry, 1999)

Thus the state equation is given by

$$\begin{aligned}
\dot{x}_1(t) &= \frac{1}{L}x_2(t) \\
\dot{x}_2(t) &= -\frac{1}{C}x_1(t) + (\frac{1}{rC} + \frac{\beta}{C})x_2(t) - \frac{\beta}{C}x_2^3(t) - \frac{1}{rC}u(t) \\
y(t) &= x_1(t) \\
x_1(0) &= x_{10} \\
x_2(0) &= x_{20}
\end{aligned} \tag{4.31}$$

We can now express the following plant model for the plant (4.31) with scaling factor $M > 1$

$$\begin{aligned}
\dot{x}_1(t) &= \frac{M}{L}x_2(t) \\
\dot{x}_2(t) &= -\frac{M}{C}x_1(t) + M(\frac{1}{rC} + \frac{\beta}{C})x_2(t) - \frac{M\beta}{C}x_2^3(t) - \frac{M}{rC}u(Mt) \\
y(t) &= x_1(t) \\
x_1(0) &= x_{10} \\
x_2(0) &= x_{20}
\end{aligned} \tag{4.32}$$

Let the constraints be $-10 \leq u(t) \leq +10\ V$, $-1 \leq \Delta u(t) \leq +1\ V$ and $-5 \leq y(t) \leq +5\ mA$.

The equilibrium point $x_{eq} = 0$ of the plant and plant model are unstable and there exists a limit cycle for all nonzero initial conditions. If the unstable limit cycle can be a stable circle with varying β, this is a sinusoidal.

For $\beta = 1, C = 1F, L = 1H$ and $r = 1Ohm$ the plant impulse response is a stable limit cycle (see Figure 4.3).

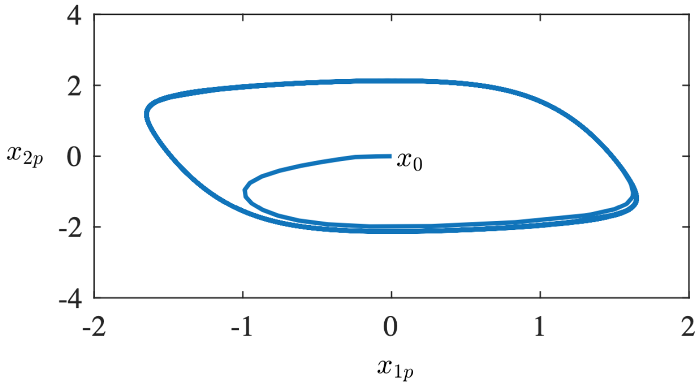

Figure 4.3 The impulse response of Van der Pol Oscillator (Sastry, 1999)

The derivatives of the output are computed until input $u(Mt)$ appears as following

$$
\begin{aligned}
\dot{y}(t) &= \dot{x}_1(t) = \frac{M}{L}x_2(t) \\
\ddot{y}(t) &= \frac{M}{L}\dot{x}_2(t) = \frac{M}{L}[-\frac{M}{C}x_1(t) + M(\frac{1}{rC} + \frac{\beta}{C})x_2(t) \\
&\quad - \frac{M\beta}{C}x_2^3(t) - \frac{M}{rC}u(Mt)] \\
&= M^2[-\frac{1}{LC}x_1(t) + (\frac{1}{rLC} + \frac{\beta}{LC})x_2(t) - \frac{\beta}{LC}x_2^3(t) \\
&\quad - \frac{1}{rLC}u(Mt)]
\end{aligned}
\tag{4.33}
$$

Since the relative degree $\gamma = 2$, there does not exist zero dynamics. The control law with the new input for the feedback linearization is given by

$$
\begin{aligned}
u(Mt) = &-rLCv(Mt) - \frac{1}{LC}x_1(t) + (\frac{1}{rLC} + \frac{\beta}{LC})x_2(t) \\
&- \frac{\beta}{LC}x_2^3(t)
\end{aligned}
\tag{4.34}
$$

Thus the second - order LTI plant model is obtained as the following

$$\ddot{y}(t) = M^2 \upsilon(Mt) \tag{4.35}$$

The new stable state equation for $M > 1$ is

$$\begin{aligned} \dot{z}_m(t) &= M \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z_m(t) + M \begin{bmatrix} 0 \\ 1 \end{bmatrix} \upsilon_m(t) \\ y_m(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} z_m(t) \end{aligned} \tag{4.36}$$

where $z \in \mathbb{R}^2$, $\upsilon_m(t) = \upsilon(Mt)$ and $\phi : x \mapsto z$ is a diffeomorphism map in a neighborhood of $x_{eq} = 0$. Note that if the linearized system is not stable, its poles can be placed on the left half plane of complex plane for stabilization or on imaginary axis of complex plane for the stable oscillation (a circle in the phase portrait) by the feedback law

$$\upsilon_m(t) = -K_1 z_1(t) - K_2 z_2(t) \tag{4.37}$$

where K_1 and K_1 are feedback coefficients for stability.

We can write the new stable state space equation for the plant with $M = 1$, as well.

$$\begin{aligned} \dot{z}_p(t) &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z_p(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \upsilon_p(t) \\ y_p(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} z_p(t) \end{aligned} \tag{4.38}$$

where $\upsilon_p(t) = \upsilon(t)$. The MPC and UMPC for the linearized plant and plant model can be implemented.

Let the continuous-time plant (4.38) must be controlled at $T_p = 0.2\ sec.$ in the plant model of the original MPC problem

$$\begin{aligned} z_p(k+1) &= \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix} z_p(k) + \begin{bmatrix} 0.02 \\ 0.2 \end{bmatrix} \upsilon_p(k) \\ y_p(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} z_p(k) \end{aligned} \tag{4.39}$$

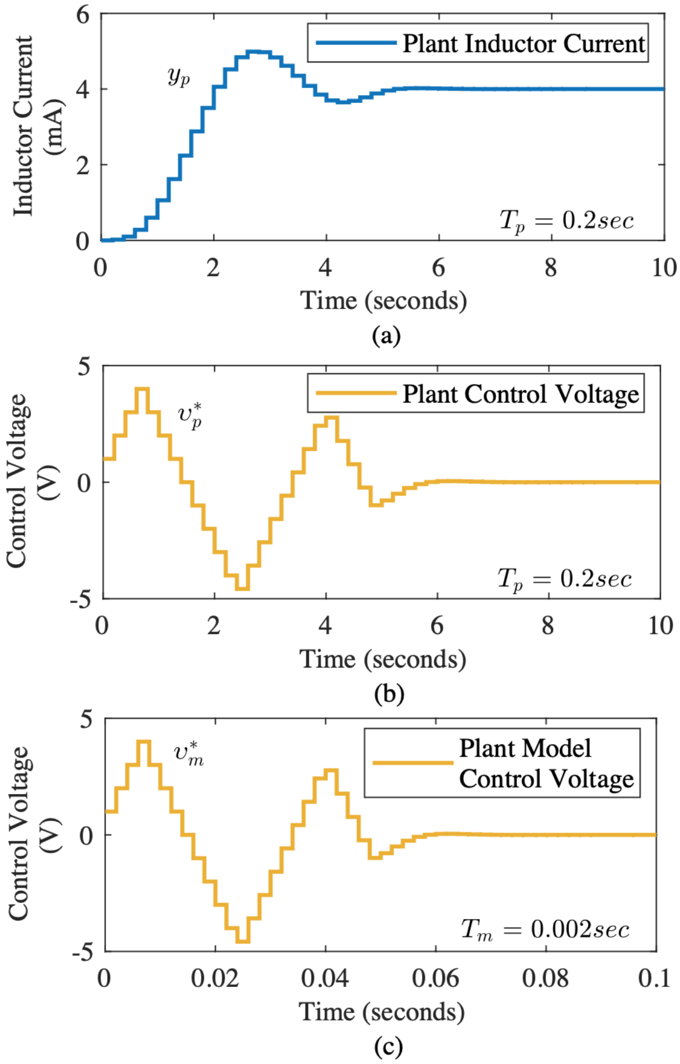

Figure 4.4 The control laws v_p^* and v_m^* for $M = 100$ (a) The MPC control law (b) The UMPC control law (Sastry, 1999)

The discrete-time form of (4.38) is the plant model of UMPC problem which has the following discrete-time plant model with $T_m = 0.002\ sec.$,

$$\begin{aligned} z_m(k+1) &= \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix} z_m(k) + \begin{bmatrix} 0.02 \\ 0.2 \end{bmatrix} v_m(k) \\ y_m(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} z_m(k) \end{aligned} \tag{4.40}$$

For the horizon $N = 8$, the matrices $q_w = 10I$, $r_w = 1$ the scaling factor $M = 100$

and the output reference value $4\ mA.$, the optimal control laws $\upsilon_p^*(k)$ and $\upsilon_m^*(k)$ of the MPC and the UMPC are simulated and plotted on Matlab MPC Toolbox (See in Figure 4.4). Thus we have free times $T_{free} = 0.198\ sec.$ to be able to implement the control law $\upsilon_m^*(k)$ to the plant at each control sampling interval $T_p = 0.2\ sec.$

When the values of the control law computed in the UMPC are stored in T_{free}, the UMPC strategy is equated to the original MPC strategy. Thus, the UMPC is used instead of the original MPC.

4.2.2 Original MPC and UMPC Applications on PVTOL Dynamics

Consider the motion equations of the planar vertical takeoff and landing (PVTOL) aircraft (plant) illustrated in Figure 4.5 (Sastry, 1999; Lehappe et al., 2016).

$$\begin{aligned}
\ddot{x}_1(t) &= -sin(x_3(t))u_1(t) \\
\ddot{x}_2(t) &= cos(x_3(t))u_1(t) - 1 \\
\ddot{x}_3(t) &= u_2(t) \\
y_1(t) &= x_1(t) \\
y_2(t) &= x_2(t)
\end{aligned} \tag{4.41}$$

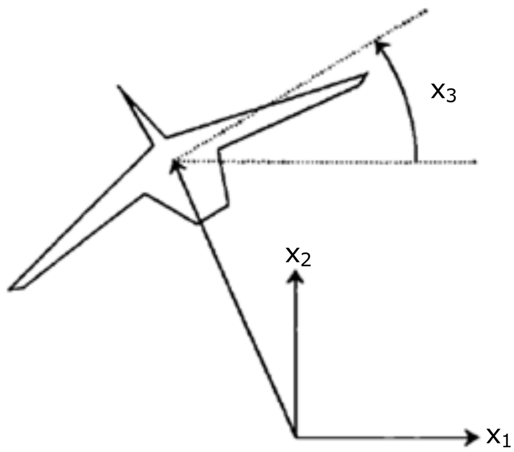

Figure 4.5 PVTOL aircraft

where $u_1(t)$ is the thrust (N), $u_2(t)$ is the rolling moment (Nm), x_1 and x_2 are the position (m) of the mass center of plant on Cartesian coordinates and the outputs of the system, x_3 is the angle (rad) of the plant relative to the x_1 - axis, "-1" is the gravitational acceleration (m/s^2) the corresponding velocities $\dot{x}_1$, $\dot{x}_2$ and $\dot{x}_3$. The aircraft has no the small coupling coefficient between the rolling moment and the lateral acceleration.

We take the derivatives on the outputs of equation (4.41) to linearize using the feedback linearization method until at least one of the inputs appears. The inputs are obtained by taking derivative twice.

$$\begin{bmatrix} \ddot{x}_1(t) \\ \ddot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -sin(x_3(t) & 0 \\ cos(x_3(t)) & 0 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{4.42}$$

The decoupling matrix is

$$T(x) = \begin{bmatrix} -sin(x_3(t) & 0 \\ cos(x_3(t)) & 0 \end{bmatrix} \tag{4.43}$$

Since $T(x)$ is singular, a static feedback control law can not be obtained. Therefore, the derivatives of the equation (4.42) are taken until $T(x)$ is nonsingular. $u_2(t)$ must come into the equation (4.42) for the nonsingular $T(x)$. Thus, we must take derivative of the equation at least twice. This approach is called *Dynamical Extension Algorithm*. The original plant (4.41) is converted to the following augmented plant.

$$\begin{aligned} \begin{bmatrix} x_1^{(4)}(t) \\ x_2^{(4)}(t) \end{bmatrix} &= \begin{bmatrix} sin(x_3(t))\dot{x}_3^2 u_1(t) - 2cos(x_3(t))\dot{u}_1(t) \\ -cos(x_3(t))\dot{x}_3^2 u_1(t) - 2sin(x_3(t))\dot{u}_1(t) \end{bmatrix} \\ &+ \begin{bmatrix} -sin(x_3(t) & -cos(x_3(t))u_1(t) \\ cos(x_3(t)) & -sin(x_3(t))u_1(t) \end{bmatrix} \begin{bmatrix} \ddot{u}_1(t) \\ u_2(t) \end{bmatrix} \end{aligned} \tag{4.44}$$

where $\ddot{u}_1(t)$ is the new input and $\dot{u}_1(t)$ and $u_1(t)$ are the new states. The new plant (4.44) will be controlled by the predictive controllers.

The new decoupling matrix

$$T(x) = \begin{bmatrix} -sin(x_3(t) & -cos(x_3(t))u_1(t) \\ cos(x_3(t)) & -sin(x_3(t))u_1(t) \end{bmatrix} \tag{4.45}$$

is nonsingular for nonzero $u_1(t)$. The matrix (4.45) does not contain zero dynamics.

The plant model of the predictive controllers for different M are given below

$$\begin{aligned}\begin{bmatrix} x_1^{(4)}(t) \\ x_2^{(4)}(t) \end{bmatrix} &= M^4 \begin{bmatrix} sin(x_3(t))\dot{x}_3^2 u_1(t) - 2cos(x_3(t))\dot{u}_1(t) \\ -cos(x_3(t))\dot{x}_3^2 u_1(t) - 2sin(x_3(t))\dot{u}_1(t) \end{bmatrix} \\ &+ M^4 \begin{bmatrix} -sin(x_3(t) & -cos(x_3(t))u_1(t) \\ cos(x_3(t)) & -sin(x_3(t))u_1(t) \end{bmatrix} \begin{bmatrix} \ddot{u}_1(Mt) \\ u_2(Mt) \end{bmatrix}\end{aligned} \tag{4.46}$$

The equation represents the original and contracted plant models of the original MPC for $M = 1$ and the UMPC for $M > 1$, respectively.

Since the equation (4.46) contains 4th order differential equations, their right sides are multiplied by M^4 to obtain the time scale solutions. Thus the equation (4.46) is linearized by the following dynamic state feedback control law

$$\begin{aligned}\begin{bmatrix} \ddot{u}_1(Mt) \\ u_2(Mt) \end{bmatrix} &= \begin{bmatrix} \dot{x}_3^2(t)u_1(t) \\ -2\dot{x}_3(t)\dot{u}_1(t)/u_1(t) \end{bmatrix} \\ &+ \begin{bmatrix} -sin(x_3(t) & cos(x_3(t)) \\ -cos(x_3(t))/u_1(t) & -sin(x_3(t))/u_1(t) \end{bmatrix} \begin{bmatrix} \upsilon_1(Mt) \\ \upsilon_2(Mt) \end{bmatrix}\end{aligned} \tag{4.47}$$

where $\upsilon_1(Mt)$ and $\upsilon_2(Mt)$ are the contracted new inputs.

Finally the linearized plant model is obtained by the control law (4.47)

$$\begin{bmatrix} x_1^{(4)}(t) \\ x_2^{(4)}(t) \end{bmatrix} = M^4 \begin{bmatrix} \upsilon_1(Mt) \\ \upsilon_2(Mt) \end{bmatrix} \tag{4.48}$$

The plant model (4.46) is converted into the equation (4.48) and used as the plant models of the predictive controllers for controlling the plant (4.46).

4.3 Numerical results discussion on nonlinear PVTOL dynamics and discussion

The augmented plant (4.48) is controlled by the original MPC that is based on the original plant model (4.48) with $M = 1$ at the sampling period $T_p = 0.4\ sec.$ The UMPC controller were also obtained for $M = 10, 25, 40$. The limit of M depends on the speed of MATLAB software installed at Intel(R) Core(TM)i5-3210M microprocessor with 2.50 Ghz and 4 GB RAM. The QP (4.20) for M was run on an active set solver using MATLAB MPC Toolbox. Note that the UMPC controllers for larger M can be run at MATLAB.

For the horizon $N = 15$, the matrices $q_w = 10I$, $r_w = I$, the input constraints $-4 \leq \upsilon_1(t) \leq 4\ N$, $-4 \leq \upsilon_2(t) \leq 4\ Nm$, the reference position $x_1 = 3\ m$ and $x_1 = 2$ m ,and $M = 1, 10, 25, 40$, the responses and optimal control laws of MPC and UMPC controllers are illustrated in Figure 4.6.

The numerical results show that as M is increased, the responses of the UMPC settle faster than the original MPC. The free sampling periods $T_{free} = 0.36, 0.39$ and $0.396\ sec.$ occurred between the control laws of the original MPC and UMPCs. In the free time periods, the settling times of the responses x_1 and x_2 are shown in Table 4.2.

The control laws of the controllers are compared in Table 4.3. As M is increased, the control law energies of the UMPC are decreased fast. This is an important advantage in control systems with high energy requirements. The computational load of the original MPC and UMPC is the same and this computation is performed in 25 iterations.

The UMPC strategy, which is adapted to the processor speed, provides fast settling and prediction responses and low control law energy requirement at each optimization sampling period.

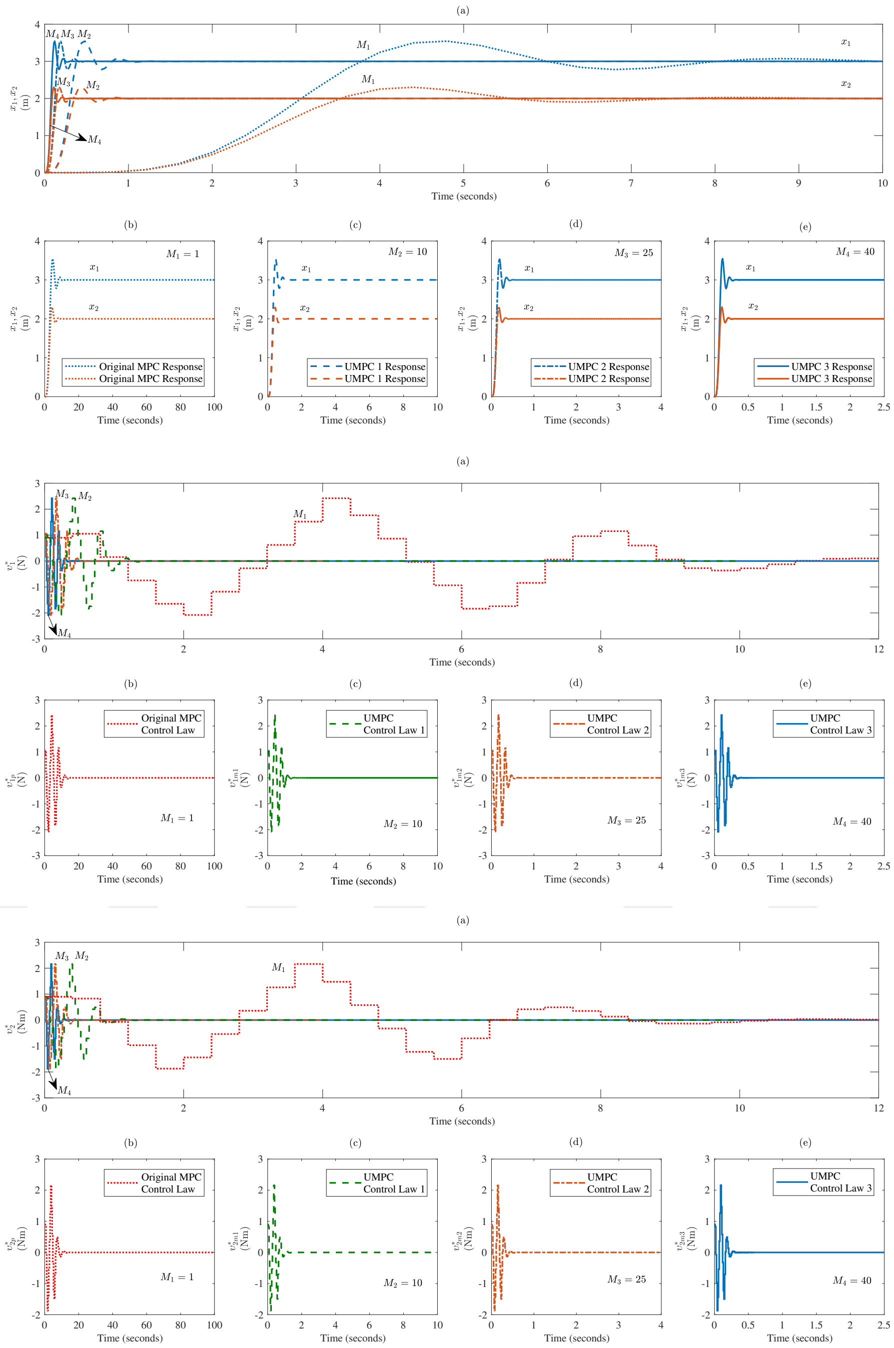

Figure 4.6 The control laws and responses of original MPC and UMPC for $M = 1, 10, 25, 40$. (a) The position responses of MPC and UMPC (b) The thrust control laws of MPC and UMPC (c) The rolling moment control laws of MPC and UMPC

The values of the control law computed in the UMPC are stored in T_{free} and the UMPC equated to the original MPC. Thus, the UMPC is used instead of the original MPC.

Table 4.2 Original MPC and UMPC responses

Controllers	Scaling Factor M	Number of Iterations	Settling Times (sec) (x_1 / x_2)
Original MPC	1	25	10 / 3.4
UMPC 1	10	25	1 / 0.34
UMPC 2	40	25	0.4 / 0.085
UMPC 3	100	25	0.1 / 0.034

Table 4.3 Original MPC and UMPC Control laws

Controllers	Free Time Sampling Period (sec)	Number of Iterations	Total Free Time until Original Settling Time (sec)	Control Energies ($v_1(t)$ / $v_2(t)$)
Original MPC	0	25	0	29.4 / 6.6
UMPC 1	0.36	25	9	2.94 / 0.66
UMPC 2	0.39	25	9.75	0.735 / 0.165
UMPC 3	0.396	25	9.9	0.294 / 0.066

4.4 Conclusion

This thesis introduces the UMPC method and an application on its PVTOL dynamics for control of a class of nonlinear dynamical systems (plant). The formulation of the original MPC is based on the linearized discrete - time model of the plant by state feedback. UMPC uses UPM, which is the contracted form of the discrete - time plant model, using an integer time scaling factor. Therefore, the UMPC strategy is a time-shifted and contracted version of the original MPC. Both use the standard MPC algorithm and have the same computational load due to online optimization. The source of the time scaling factor is the speed of the microprocessor used. UMPC is adapted to speeds of microprocessors. Thus, the UMPC provides the following contributions

- Fast rise and settling times,
- Fast prediction in each optimization period,
- Low control energy,

- Eliminating the problem of slow working due to the online computational load of the original MPC and enabling the application of MPC algorithms to fast dynamics,
- Incorporating different control scenarios and the other control techniques since the free time periods occur between the original MPC and UMPC due to time-shifting,
- Improving the methods in the references.

CHAPTER 5

CONCLUSIONS

When the research topics for the MPC algorithms in the literature are examined, there are closed - loop stability, feasbility and slow operation issues. It has been seen that the numerous studies have been conducted to solve these problems. However the real - time explicit, implicit MPC and Model Reduction techniques developed to apply theoretical MPC (nominal, robust, target tracking linear, nonlinear and stochastic MPC) algorithms are successful under some important limitations, which can be applied to the small - size optimization problems due to the speed and storage requirements, but cannot completely solve the issues.

The main focus of this thesis is to speed up the slow response problems caused by online computation load of the MPC algorithms. In our study, the Nominal MPC strategy for reducing the online computation load is compressed adapting it to the speed of the processors used instead of making improvements on the structure of the MPC optimization problem or by decreasing dimension of the plant model. This modification is done by using an integer time scaling factor. The free time periods are created free times between the control laws of the nominal MPC (the original MPC) and the modified nominal MPC (UMPC) and they are the time - shifted version of each other. The energy of control law computed in UMPC is lower than the control law of the nominal MPC. Since UMPC computes the control law on the UPM, it results the fast rise and settling times and predictions at each online optimization sampling periods in UMPC, which is the same as the responses of the original MPC. The fast - computed control law is dilated by the free times by storing its values at each sampling period and are equated to the original control law. The equated control law is applied to the plant and this procedure is repeated until the new state measurements from the plant become. In Chapters 3 and 4, the method is applied on the linear and nonlinear plant models and the contributions are quantized and illustrated on the numerical examples. The results prove that the UMPC solves the slow operation problems of the MPC algorithms and can be applied to the fast - sampled plants by exploiting speeds of microprocessor.

Another noteworthy aspect of the study is the free time period. It can be used for the improvements of the existing MPC algorithms. In this way,

- The prediction capability of MPC can be increased online during these time periods,
- Plant - plant model mismatches from disturbance or unforeseen situations can be compensated by using online system identification methods,
- Adaptive control or other control techniques can be integrated with UMPC,
- The main problems of closed - loop stability and feasibility can be overcome with some improvements to be made during these periods.

This property of the study will provide a motivation on subsequent studies for controlling stochastic and nonlinear dynamical systems and we will tend on the above improvements and issues in our next studies.

REFERENCES

Bemporad, A. (2011). Automatic control. In *Lecture notes in University of Trento*, 1–31.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*(1), 3–20.

Bertsekas, D. (2005). Nonlinear programming. *Lecture notes in MIT*, 1–23.

Boom, V., J.J., T., & Backx, T. (2007). Model predictive control. *Lecture notes in TU Delft*, 1–41.

Borrelli, F., Baoti´c, M., Pekar, J., & Stewart, G. (2010). On the computation of linear model predictive control laws. *Automatica*, *46*(6), 1035–1041.

Borrelli, F., Bemporad, A., & Morari, M. (2015). *Predictive control for linear and hybrid systems*. University of Berkeley.

Boyd, S. & Vandenberghe, L. (2004). *Convex optimization*, (1st ed.). (365-440). UK: Cambridge University Press.

Boyd, S. & Vandenberghe, L.(2009).*Convex Optimization* (455-623). UK:Cambdrige University Press.

Bubnicki, Z. (2005). *Modern vontrol theory*. Wroclaw: Springer.

Burke, J. (2020). Nonlinear continuous optimization. *In lecture notes in University of Washington*, 1–110.

Cai, X., Tippett, M., Xie, L., & Bao, J. (2014). Fast distributed mpc based on active set method. *Computers & Chemical Engineering*, *71*, 158–170.

Chen, C. (1999). *Linear system theory and design*. Oxford: Oxford University Press.

Chen, D. (2016). *Optimal control of continuous and discrete time systems via generating functions*. PhD thesis, Nagoya University, Japan.

Diehl, M. (2015). Direct single and direct multiple shooting. *Lecture Notes in University of Freiburg*, 1–21.

Diehl, M., Bock, H., Diedam, H., & Wieber, P. (2009). Fast direct multiple shooting algorithms for optimal robot control. *Fast Motions in Biomechanics and Robotics*, 1–28.

Diehl, M. & Gros, S. (2017). *Numerical optimal control (preliminary and incomplete draft)*. Freiburg: University of Freiburg.

Dundar, B. & Cihanbegendi, O. (2019). Application of a fast model Predictive control method. *In 2019 International Symposium on Multidisciplinary Academic Studies (IMASES)*, 169–185. Izmir: IMASES.

Dundar, B. & Cihanbegendi, O. (2020). A fast strategy for model predictive control. *International Journal of Scientific Research and Management*, *8*(3), 1–19.

Dunlap, D., Collins, J., Emmanuel, G., & Caldwell, C. (2008). Sampling based model predictive control with application to autonomous vehicle guidance. In *Florida Conference on Recent Advances in Robotics*, 1–6. Florida: FCRAR.

Elixmann, D. (2016). *Economic model - predictive control of membrane bioreactors for wastewater treatment*. Ph.D thesis, Aachen University, Aachen.

Fletcher, R. (2013). *Practical methods of optimization*. Cambridge: Wiley.

Forsgren, A., Gill, P., & Wright, M. (2002). Interior methods for nonlinear optimization. *SIAM Review*, *44*(4), 525–597.

Garcia, C., Prett, D., & Morari, M. (1989). Model predictive control: theory and practice — a survey. *Automatica*, *25*(3, 335–348.

Ghaoui, L. (2012). Convex optimization and applications. *Lecture notes in University of Berkeley*, 1–9.

Gordon, G. & Tibshirani, R. (2012). Newton's method under equality constrains. *Lecture notes in Carnegie Mellon University*, 1–6.

Ha, S. (2017). PMP v.s. Bellman's dynamic programming. *Lecture notes in Department of Mathematical Sciences of Seoul National University*, 1–56.

Hafez, A., Iskandarani, M., Givini, S., Yousefi, S., Rabbath, C., & Beaulieu, A. (2015). Using linear model predictive control via feedback linearization for dynamic encirclement. *In 2015 Proceedings of the American Control Conference*, 3868–3873, Cape Town: ACC.

Hedengren, J. (2020). *Optimization methods*. Retrieved February 15, 2020 from http://www.apmonitor.com/me575/uploads/Main/interior_point_lecture.pdf.

Hedrick, J. & Girard, A. (2005). *Control of nonlinear dynamic systems: theory and applications*. Michigan: University of Michigan.

Hein, M. (2010). Convex optimization and modeling. *Lecture notes in Saarland University*, 1–41.

Jerez, J., Ling, K., Constantinides, G., & Kerrigan, E. (2011). MPC for deeply pipelined FPGA implementation: algorithms and circuitry. *Imperial College London, UK, Nanyang Technological University, Singapure*, 1–29.

Johansson, M. (2015). Control theory and practice. *Lecture notes in KTH*, 1–9.

Kall, P. (2015). *Computational economics and finance part v, constrained optimization, Univerity of Zurich*. Retrieved January 5, 2020 from Retrieved January 5, 2020 from http://www.business.uzh.ch/dam/jcr: ffffffff-cd5d-ce16-ffff-fffff84b97f2/5ConstrainedOptimization.pdf.

Katja, M. & Felis, M. (2011). Optimization in robotics & biomechanics. *Tutorial in IWR, University of Heidelberg*, 1–72.

Kirk, D. (2004). *Optimal control theory an introduction*. New York: Dover.

Koehler, S., Danielson, C., & Borrelli, F. (2017). A primal - dual active - set method for distributed model predictive control. *Optimal Control Applications and Methods*, *38*(3),399–419.

Kouvaritakis, B., Basil, & Cannon, M. (2016). *Model predictive control*. Oxford: Springer.

Lars, S. (2018a). Equality optimization, modern optimization techniques. *Lecture notes in University of Hildesheim*, 1–39.

Lars, S. (2018b). Inequality constrained optimization, modern optimization techniques. *Lecture notes in University of Hildesheim*, 1–30.

Lehappe, V., Aoustion, Y., Martinez, L., & Moog, C. (2016). Partial linearization of the PVTOL aircraft with internal stability. *HAL*, 1–7.

Ling, K. & Maciejowski, J. (2009). A comparison of interior point and active set methods for FPGA implementation of model predictive control. *In 2009 Proceedings of the Europen Control Conference*, 1–7.

Ling, K., Wu, B., & Maciejowski, J. (2008). Embedded model predictive control (mpc) using a fpga. *In 2008 Proceedings of the 17th World Congress The International Federation of Automatic Control*, 41(2), 15250–15255. Seoul: IFAC.

Ling, K., Yue, S., & Maciejowski, J. (2006). A FPGA implementation of model predictive control. In *In 2006 Proceedings of the American Control Conference*, 1–7.

Liu, Z., Xie, L., Bemporad, A., & Lu, S. (2018). Fast linear parameter varying model predictive control of buck DC-DC converters based on FPGA. *IEEE Access*, *6*, 52434–52446.

M. Zeilinger, C. Jones, F. B. & Morari, M. (2016). Model predictive control part i - introduction. *Lecture notes in Institut fur Automatik, ETH Zurich, Institute for Dynamic Systems and Control, ETH Zurich, UC Berkeley, EPFL*, 1–158.

MathWorks, Inc. (2017). *Matlab software.* R2017a 9.2.0.538062.

Moor, B. (2017). Introduction to model predictive Control (MPC), ESAT, KU Leuven. *Lecture notes in ESAT, KU Leuven*, 1–200.

Morari, M. (2013). Fast model predictive control (MPC). *Lecture Notes in Automatic Control Laboratory, ETH Zurich*, 1–80.

Orchi, T., Mahmud, M., & Oo., A. (2018). Partial feedback linearizing model predictive vontrollers for multiple photovoltaic units connected to grids through a Point of common coupling. *Electronics, MDPI*, 1–18.

Paiva, L. (2014). *Numerical methods for optimal control and Model predictive control, Porto*. PhD thesis, University of Porto, Porto.

Pena, D., Alamo, T., Bemporad, A., & Camacho, E. (2004). A Dynamic programming approach for determining the explicit solution of linear MPC controllers. *IEEE*, 2479–2484.

Primbs, J., c, V. N., & Doyle, J. (1999). nonlinear optimal control: a control lyapunov function and rceding horizon perspective. *In 1999 Asian Journal of Control,*, *1*(1), 14–24.

Qi, R., Rongbin, Mei, H., Chen, C., & Qian, F. (2015). A fast MPC algorithm for reducing computation burden of MIMO. *Chinese Journal of Chemical Engineering*, *23*(12), 2087–2091.

Qin, S. & Badgwell, T. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, *11*(7), 733–764.

Rao, A. (2015). A survey of numerical methods for optimal control. *Tutorial in Department of Mechanical and Aerospace Engineering, University of Florida*, 1–33.

Rao, C., Wright, S., & Rawlings, J. (1998). Application of interior - point methods to model predictive control. *Journal of Optimization Theory and Applications*, *99*(3), 723–757.

Ratkovic, K. (2016). Limitations in direct and indirect methods forsolving optimal control problems in growth theory. *Industrija*, *44*(4),19–46.

Rawlings, J., Mayne, D., & Diehl, M. (2019). *Model predictive control, theory, computation, and design*, (2nd ed.). London: Nob hill.

Robert, J. (2013). Nonlinear optimization, the interior - point algorithm, nonlinear optimization, lecture notes. *Lecture notes in Princeton University*, 1–14.

Sastry, S. (1999). *Nonlinear systems analysis, stability and control*. Berkeley: Springer.

Schnelle, F. & Eberhard, P. (2015). Constraint mapping in a feedback linearization / MPC scheme for trajectory tracking of underactuated multibody systems. *In 2015 International Federation of Automatic Control*, *48*(23), 446–451. Seville: IFAC.

Seborg, D., Mellichamp, D., Edgar, T., & Francis, J. (2010). *Process dynamics and control*, (3rd ed.). California: Wiley.

Shahzad, A., Kerrigan, E., & Constantinides, G. (2010). A warm - stad interior - point method for predictive control. *Department of Electrical and Electronic Engineering, Imperial Colloge London*, 1–6.

Singh, A. & Ravikumar, P. (2017). Primal - dual interior - point methods, convex optimization. *Handout in Department of Statistics at Carnegie Mellon University*, 1–19.

Stengel, R. (2010). *Optimal control and estimation*. New York: Dover.

Taha, A. (2015). Introduction to optimal control. *Tutarial in UTSA*, 1–23.

Taha, A. (2017). Optimization, optimal control, and model predictive control constrained control of dynamic systems. *Tutarial in UTSA*, 1–32.

Tibshiran, R. (2017). Newton's method. *Lecture notes in Department of Statistics at Carnegie Mellon University*, 1–24.

Tibshirani, R. (2017). Barrier method. *Lecture notes in Department of Statistics at Carnegie Mellon University*, 1–23.

Tits, A. (2018). Optimal control. *Lecture notes in University of Maryland,*, 1–202.

TøNdel, P., Johansen, T., & Bemporad, A. (2003). An algorithm for multi - parametric quadratic programming and explicit mpc solutions. *Automatica*, *39*(3), 489–497.

Wang, J. & Boyd, S. (2010). Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, *18*(2), 267–278.

Wang, L. (2008). *Model predictive control system design and implementation using MATLAB*. Melbourne: Springer.

Wolfram Alpha LCC (2020).

Xie, S., Hu, X., Xin, Z., & Brighton, J. (2018). Pontryagin's minimum principle based model predictive control of energy management for a plug - in hybrid electric bus. *Elsevier*, *236*, 893–905.

Yildirim, E. & Wright, S. (2002). Warm - start strategies in interior - point methods for linear programming. *SIAM Journal on Optimization*, *12*(3), 782–810.

Zak, S. (2013a). Model - based predictive control. *Lecture notes in Purdue University*, 1–27.

Zak, S. (2013b). Pontryagin's minimum principle. *Lecture notes in Purdue University*, 1–6.

Zannon, M. (2015). *Efficient conlinear model predictive control formulations for economic objectives with aerospace and automotive applications*. Ph.D thesis, KU Leuven, Belgium.

Zeilinger, M. (2011). *Real - time model predictive control*. PhD thesis, University of Stuttgart, Stuttgart.

Zeilinger, M. & Jones, C. (2011). Robust stability properties of soft constrained MPC robust stability properties of soft constrained MPC. *IEEE*, 1–9.

Zhang, Y., Nguyen, H., Naidu, D., Zou, Y., & Cai, C. (2013). Time scale analysis and synthesis for model predictive control. *In 2013 Proceedings of the WSEAS–NAUN 4th International Conference on Circuits, Systems, Control, Signals*, 27–32. Valencia: CSCS.

APPENDICES

APPENDIX-1: Time Scale Separation on Contracted Plant Models

Consider the following controllable plant

$$\begin{aligned}\frac{dx_p(t)}{dt} &= Ax_p(t) + Bu_p(t)\\ y_p(t) &= Cx_p(t)\\ x_p(t_0) &= x_0 = \begin{bmatrix} x_{01} & x_{20}\end{bmatrix}^T, \quad \forall \ t \geq t_0\end{aligned} \tag{1}$$

where state variable $x_p(t) \in \mathbb{R}^n$, initial state variable $x_0 \in \mathbb{R}^n$, control variable $u_p(t) \in \mathbb{R}^r$, output variable $y_p(t) \in \mathbb{R}^s$.

Let plant 1 has a two - time - scale sub linear systems.

$$\begin{aligned}\frac{dx_{p1}(t)}{dt} &= A_1x_{p1}(t) + B_1u_{p2}(t)\\ y_{p1}(t) &= C_1x_{p1}(t)\\ x_{p1}(t_0) &= x_{20}, \quad \forall \ t \geq t_0\end{aligned} \tag{2}$$

$$\begin{aligned}\frac{dx_{p2}(t)}{dt} &= A_2x_{p2}(t) + B_2u_{p2}(t)\\ y_{p2}(t) &= C_2x_{p2}(t)\\ x_{p2}(t_0) &= x_{20}, \quad \forall \ t \geq t_0\end{aligned} \tag{3}$$

where x_{p1} and x_{p2} are $\ell-$ and $n-\ell$ - dimensional slow and fast state and output vectors; u_{p1} and u_{p2} are $m-$ and $s-m$ - dimensional slow and fast state and output vectors, respectively; y_{p1} and y_{p2} are $p-$ and $s-\ell$ - dimensional slow and fast state and output vectors, respectively. The system A_1, A_2, the control B_1, B_2 and output C_1, C_2 matrices are the corresponding the dimensions of the matrices A, B and C, respectively. $u_p(t)$ is composite of the sub control variables given by

$$u_p(t) = u_{p1}(t) + u_{p2}(t) \tag{4}$$

The contracted dynamics with $M > 1$ based on the sub plants (2) and (3) can be written as

$$\begin{aligned} \frac{dx_m(t)}{dt} &= M(A_1 x_{m1}(t) + B_1 u_{m1}(t)) \\ y_{m1}(t) &= C_1 x_{m1}(t) \\ x_{m1}(t_0) &= x_{10}, \quad \forall \ t \geq t_0 \end{aligned} \tag{5}$$

$$\begin{aligned} \frac{dx_{m2}(t)}{dt} &= M(A_2 x_{m2}(t) + B_2 u_{m2}(t)) \\ y_{m2}(t) &= C_2 x_{m2}(t) \\ x_{m2}(t_0) &= x_{20}, \quad \forall \ t \geq t_0 \end{aligned} \tag{6}$$

where state variables $x_{m1}(t) \in \mathbb{R}^n$ and $x_{m1}(t) \in \mathbb{R}^{n-\ell}$, the measured or estimated initial state variables x_{10} and x_{20} from the sub plants (2) and (3), control variables $u_{m1}(t) \in \mathbb{R}^m$ and $u_{m2}(t) \in \mathbb{R}^{r-m}$, output variables $y_{m1}(t) \in \mathbb{R}^p$ and $y_{m2}(t) \in \mathbb{R}^{s-p}$ of the dynamics (3) and (4).

Let the sub plants and the contracted sub plants be sample and controlled at T_{p1} and T_{p1} and at $T_{m1} = T_{p1}/M$ by the original MPC controlers and $T_{m2} = T_{p2}/M$ by the UMPC controllers, respectively, under $u_p(t) = u_m(Mt)$. That holds the condition (3.95). In that case, the time scale separation - MPC method is speed up by the time scale - separation UMPC controllers.

APPENDIX-2: Zero Dynamics

The input - output control law (4.8) can satisfy the closed - loop stability of the linear system with transfer function $(M/s)^{\gamma}$, which has γ - integrator from υ to y. In that case, $(n-\gamma)$ of the state variables are unobservable by state feedback. Note that the observable states does not affect controllability. There exists a diffeomorphism map $\overline{\phi}$ in a neighborhood of x_{eq} that is given by

$$\overline{\phi} : x \mapsto \begin{bmatrix} z \\ \xi \end{bmatrix} \tag{7}$$

where $z \in \mathbb{R}^{\gamma}$ and $\xi \in \mathbb{R}^{n-\gamma}$ represents unobservable internal dynamics. Thus, the feedback linearized plant model with internal dynamics are given by

$$\begin{aligned} \dot{z}(t) &= M(Az(t) + B\upsilon(Mt)) \\ \dot{\xi} &= \psi(z, \xi) \\ y(t) &= z_1(t) \end{aligned} \tag{8}$$

where $\psi(.)$ is an internal vector field. $z(t)$ does not depend on ξ and are not a function of input $u(t)$. Internal dynamics must analyze but generally the analysis is difficult. Output $y(t)$ depends on $z(t)$. If $y(t) \equiv 0$, internal dynamics become

$$\dot{\xi} = \psi(0, \xi) \tag{9}$$

The problem (8) is called *Output - Zeroing Problem* and called *Zero Dynamics*. Note that zeros of transfer function for linear dynamical system are poles of zero dynamics. The zero dynamics can be analyzed in a neighbourhood of x_{eq}. If the equilibrium point of the zero dynamics is locally asymptotically (exponentially) stable, (8) is locally asymptotically (exponentially) *Minimum Phase*. If zero dynamics are stable, the feedback linearization method of nonlinear systems can be used.

There are two methods to be able to make stabilization of unstable internal dynamics. These are

- to redefine output $y(t)$,
- to use input - state linearization.

Feedback linearization technique cannot be used for all nonlinear systems, the full state has to be measured for its success and no robustness is guaranteed in the presence of parameter uncertainty or unmodeled dynamics (Hedrick & Girard, 2005; Sastry, 1999).