Bit-stream Based Predictive Controllers for Linear and Nonlinear systems

Bo Yu

A thesis submitted of the requirements for the degree of Master of Engineering in Electrical Engineering,
The University of Auckland, 2015.

ABSTRACT

Controller design based on model predictive controller (MPC) is being widely applied in industry and studied by academia during past few decades. In recent years, with the wide application of networked control systems, bit-stream based control design method is getting increasing attention from researchers. The present study essentially focus on designing continuous time MPC and implement them in bit-stream environment. This is called as bit-stream MPC. The performance of bit-stream based MPC is investigated both via simulations and experiments.

The study begins with a review of model predictive control in discrete time domain and the bit-stream technique. To successfully implement bit-stream controllers, several functions are initially implemented in MATLAB which can convert the analog or multi-bit digital signals into single bit. Then the discrete time MPC is adopted with bit-stream technique. Although the discrete time MPC has been very popular amongst practitioners during the past few decades, the controller still has some disadvantages such as choice of sampling time and numerical sensitivities.

To overcome the limitations and difficulties associated with discrete time controllers, continuous time approach to controller design was preferred. Therefore, the next part of the research begins with designing controllers in continuous time domain. Initially continuous time model predictive controller

(CMPC) was designed for linear systems based on the state space model of the system and then it is combined with bit-stream technique.

In practice most of the systems are nonlinear to some extent. Therefore the next part of the study focuses on the design of CMPC for nonlinear systems (NLCMPC) based on state space models of the system. And also bit-stream technique is used on the NLCMPC.

The last phase of the research deals with hardware implementation of bit-stream based CMPC using HILINK. An experimental prototype of DC servo motor has been considered for such implementation. The performance of bit-stream based linear CMPC has been implemented using HILINK and the tracking performance of such controllers is investigated by considering different types of references.

ACKNOWLEDGEMENTS

It takes me a whole year to work and finish this master thesis. This inspiring, exciting, challenging and memorable period brought me infinite passion and achievement. I would like to take this opportunity to express my sincere gratitude to all those who gave me guidance, encouragements, inspirations and supports which helped me overcome all the difficulties I have met.

Firstly, I would like to express my deep and sincere gratitude to my respected supervisor: Dr. Akshya Swain for his careful guidance, education and comments throughout my research. He always made himself available despite his really busy schedules. It has been a privilege to participate in this research under his supervision.

All my friends and lab buddies made great contributions to my work. In particular, I would like to extend my indeed thanks to one of my dearest friends, also my upperclassman, Dhafer Almakhles for his support and help during my university life. I am very fortunate to have this friendship.

Finally, I would like to my deep appreciate to my Mum and Dad for their endless love, limitless patience and support throughout my life.

Thanks for all my friends, classmates and the university staff, this thesis could not be completed without any of them.

CONTENTS

Chapter 1

Introduction

1.1 Motivation

Model predictive control (MPC) has made a significant impact on industrial process control systems since it was originated in the late seventies. It was brought up to solve the existing problems of the traditionally used self-tuning control such as lacking robustness [1, 2]. With over forty years' development, model predictive control gradually becomes the most important approach to the advanced control of complex industrial processes [2, 3, 4]. Compare to the traditional controllers such as the conventional PID controller, MPC has two crucial advantages:

a) The working principle of MPC is easy to be understood. MPC works like a human who predicts the future outcome and choose the suitable actions at the present in order to obtain a satisfactory output over some horizon in the future.

b) Although there are several types of MPC which fit to different kinds of systems, the key characteristics of all of them are essentially similar. Thus different kinds of plants could easily adopt MPC controller with little modification from the basic formulation of MPC. This makes MPC a much wider range of applications.

However, traditional MPC is too complex to implement in real-time embedded systems. Especially recent years, networked control systems are widely applied in industrial field. In a networked control system there may be very limited resources. In this situation bit-stream technique can be used to design controller due to various advantages this offer.

It converts either analog or multi-bits digital signal into bit-stream (single-bit) output through a Delta-Sigma (Δ-Σ) modulator. Bit-stream signal processing is mainly proposed to reduce the silicon consumption and the physical areas for routing bit-parallel signals in digital integrated circuits such as FPGA and VLSI. Furthermore, bit-stream technique reduces the number of interface channels between the subsystems from multiple to single channel, thereby consumes significantly less hardware resources compared to traditional multibit processing. In control and power electronics applications, a pulse width modulator (PWM) is not needed anymore since the bit-stream signal is like a fine-grained PWM and hence, it can derive DC-DC converters directly.

1.2 Objectives

Motivated by the success of model predictive controllers and the advantage of bit-stream based technology, the research carried out in this study intends to achieve the following objectives:

i) Carry out a comprehensive review of model predictive control and bit-stream based technology, apply both MPC and bit-stream based

MPC on a specific system and compare the performance of different controllers, discuss their stability and tracking ability.

ii) Investigate the performance of continuous time MPC for classic nonlinear systems, apply the controller to specific system with 不 bit-stream technology.

iii) Design continuous time model predictive controllers observers for complex nonlinear systems and study the performance of bit-stream based MPC.

iv) Implement the continuous time MPC to control a real DC motor using a servo motor rig.

The organization of this thesis proceeds as follows: chapter 2 presents a comprehensive review of model predictive control algorithms in discrete time, apply MPC on specific system with bit-stream modulator and discuss their performance. Chapter 3 studies the principles of linear continuous time MPC and apply it with bit-stream modulator with simulation. In chapter 4, design methods of linear continuous time MPC is extended for classic nonlinear systems and a bit-stream modulator is used together with this controller to investigate the performance. In chapter 5, the continuous time model predictive controller is realized in hardware with implementation on a servo DC motor rig. Finally, the conclusions of the thesis and some possible directions for future investigation on this research area were presented in Chapter 6.

Chapter 2

Bit-Stream Based Discrete Time MPC Control

2.1 Introduction

Model Predictive Control (MPC) computes a trajectory of a future manipulated variable *u* to optimize the future behavior of the system output *y*. It clearly computes the predictive behavior over some horizon while most classical control laws, e.g. PID, do not consider the future influences of current control actions.

Before applying the bit-stream modulator to wider range, it should be adopted on the discrete time MPC first in order to investigate its performance step by step. This chapter comprehensively review the algorithm of model predictive control from a single-input-single-output (SISO) simply system extending to a multiple-input-multiple-output (MIMO) complex system. After that, such controllers are applied on an isolated thermal system with simulation to investigate the performance of both MPC and bit-stream based MPC.

In the meanwhile, bit-stream converter and inverter are developed in MATLAB/Simulink in order to apply it on other types of systems conveniently. And in the further simulation, such modulators are used directly as a blocked model without building new models.

2.2 Model Predictive Control

2.2.1 Control law design

Consider a system which admits a locally linear model:

$$A\left(z^{-1}\right)y(t)=z^{-d}B\left(z^{-1}\right)u(t-1)+x(t) \tag{2.1}$$

where A and B are polynomials in the backward shift operator

$$A\left(z^{-1}\right)=1+a_1z^{-1}+\cdots+a_{n_a}z^{-n_a} \tag{2.2}$$

$$B\left(z^{-1}\right)=b_0+b_1z^{-1}+\cdots+b_{n_b}z^{-n_b} \tag{2.3}$$

d=dead time of the system
$u(t)$=control input
$y(t)$=output

x(t) is the noise or uncertainty in the model and can considered to be of the form

$$x(t)=\frac{C\left(z^{-1}\right)}{1-z^{-1}}\xi(t) \tag{2.4}$$

where $\xi(t)$ is a random noise with zero mean which disturbs the system but it is not measurable. C is a known polynomial:

$$C\left(z^{-1}\right)=1+c_1z^{-1}+\cdots+c_{n_c}z^{-n_c} \tag{2.5}$$

Define $\Delta=1-z^{-1}$ then:

$$A\left(z^{-1}\right)\Delta y(t)=z^{-d}B\left(z^{-1}\right)\Delta u(t-1)+C\left(z^{-1}\right)\xi(t) \tag{2.6}$$

At each time t, MPC minimizes the following the cost function:

$$J(N_1,N_2,N_u)=E\left[\left(Q_1+Q_2\right)\right] \tag{2.7}$$

$$Q_1=\sum_{j=N_1}^{N_2}\delta(j)\left[\hat{y}(t+j)-w(t+j)\right]^2,\ Q_2=\sum_{j=1}^{N_u}\lambda(j)\left[\Delta u(t+j-1)\right]^2$$

where N_1 = minimum costing horizon

N_2 = maximum costing horizon

N_u = control horizon

$\sigma(j), \lambda(j)$ = weighting sequences

$w(t+j)$ = future set-points presumed to be known

The first step to optimize the cost function is to get the optimal prediction of *y(t+j)* while $N_1 \le j \le N_2$.

And

$$A(z^{-1})\Delta y(t) = z^{-d}B(z^{-1})\Delta u(t-1) + C(z^{-1})\xi(t) \tag{2.8}$$

Thus

$$y(t) = z^{-d}\frac{B}{A}u(t-1) + \frac{C}{A\Delta}\xi(t) \Rightarrow y(t+j) = \frac{B}{A}u(t+j-d-1) + \frac{C}{A\Delta}\xi(t+j) \tag{2.9}$$

Then consider the immeasurable part:

$$\frac{C}{A\Delta}\xi(t+j)$$

Assume

$$\frac{C}{A\Delta} = E_j + z^{-j}\frac{F_j}{A\Delta} \tag{2.10}$$

where $E_j(z^{-1}) = e_0 + e_1 z^{-1} + \dots\dots + e_{j-1}z^{-(j-1)}$

$F_j(z^{-1}) = f_0 + f_1 z^{-1} + \dots\dots + f_{n_f} z^{-n_f}$

$n_f = \max(n_1 - 1, n_2 - j) = \max(n_a, n_2 - j)$

$n_1 = \text{maximum degree of } A\Delta = n_a + 1$

$n_2 = \text{maximum degree of } C$

Thus we could get

$$z^j E_j A\Delta y(t) = z^j E_j z^{-d} B\Delta u(t-1) + z^j E_j C\xi(t) \tag{2.11}$$

In the meanwhile,

$$E_j A\Delta = C - z^{-j}F_j$$

So

$$\begin{aligned}
&E_j A\Delta y(t+j)=E_j B\Delta u(t+j-d-1)+E_j C\xi(t+j)\\
\Rightarrow\ &\left(C-z^{-j}F_j\right)y(t+j)=E_j B\Delta u(t+j-d-1)+E_j C\xi(t+j)\\
\Rightarrow\ &Cy(t+j)=E_j B\Delta u(t+j-d-1)+F_j y(t)+E_j C\xi(t+j)\\
\Rightarrow\ &y(t+j)=\frac{1}{C}\left[E_j B\Delta u(t+j-d-1)+F_j y(t)+E_j C\xi(t+j)\right]
\end{aligned}\tag{2.12}$$

For $C\left(z^{-1}\right)=1$, the equation becomes

$$\begin{aligned}
y(t+j)&=E_j B\Delta u(t+j-d-1)+F_j y(t)+E_j\xi(t+j)\\
&=G_j\Delta u(t+j-d-1)+F_j y(t)+E_j\xi(t+j)
\end{aligned}\tag{2.13}$$

The prediction equation would be

$$\begin{aligned}
\hat{y}(t+j)&=E_j B\Delta u(t+j-d-1)+F_j y(t)\\
&=G_j\Delta u(t+j-d-1)+F_j y(t)
\end{aligned}\tag{2.14}$$

Where $G_j=E_jB=g_{j,0}+g_{j,1}z^{-1}+.....+g_{j,n_g}z^{-n_g}$, $n_g=n_b+j-1$

Because the system dead time is *d*, then the output will be influenced by *u(t)* after sampling period d+1. And to optimize the cost function, the set of control signals *u(t), u(t+1),…, u(t+N)* needs to be obtained.

Now from the model of future output, the minimum value would be $j=N_1=d+1, N_2=d+N$ and $N_u=N$, for *j=d+1, d+2,…, d+N*, the output equations would be

$$\begin{aligned}
&y(t+d+1)=G_{d+1}\Delta u(t)+F_{d+1}y(t)+E_{d+1}\xi(t+d+1)\\
&y(t+d+2)=G_{d+2}\Delta u(t+1)+F_{d+2}y(t)+E_{d+2}\xi(t+d+2)\\
&\cdots\cdots\\
&\cdots\cdots\\
&y(t+d+N)=G_{d+N}\Delta u(t+N-1)+F_{d+N}y(t)+E_{d+N}\xi(t+d+N)
\end{aligned}\tag{2.15}$$

Analyze the different terms of *y(t+j):*

$F_{d+1}y(t)=\left(f_0+f_1z^{-1}+....+f_{n_f}z^{-n_f}\right)y(t)$ is dependent on the past values of y(t) $\Rightarrow$ known

$E_{d+1}\xi(t+d+1)$ is dependent on both past and future values of the noise ξ $\Rightarrow$ unknown

Consider the term $G_j\Delta u(t+j-d-1)$. This term can be split into two parts, one is dependent on the past values of *u(t)* (known) and the other is dependent on the future values of *u(t)* (unknown).

For *j=d+1*:

$$
\begin{aligned}
G_{d+1}(z^{-1})\Delta u(t+d+1-d-1) &= G_{d+1}(z^{-1})\Delta u(t) \\
&= \left[g_{d+1,0}+g_{d+1,1}z^{-1}+....+g_{d+1,n_g}z^{-n_g}\right]\Delta u(t) \\
&\quad \textit{where } n_g = n_b + j - 1 = n_b + d \\
&= \underbrace{g_{d+1,0}\Delta u(t)}_{\textit{Unknown at time } t} + \underbrace{g_{d+1,1}\Delta u(t-1)+.....+g_{d+1,n_g}\Delta u(t-n_g)}_{\textit{known at time } t} \\
&= \underbrace{g_0\Delta u(t)}_{\textit{Unknown at time } t} + \underbrace{\left[G_{d+1}(z^{-1})-g_{d+1,0}\right]z\,\Delta u(t-1)}_{\textit{known at time } t} \\
&\quad \textit{Note that } g_{ij}=g_j \;\; \textit{for } \; j=0,1,2,..<i
\end{aligned}
$$

Also we can get that for *j=d+2* to *j=d+N*:

$$
\begin{aligned}
G_{d+2}(z^{-1})\Delta u(t+d+2-d-1) &= \underbrace{g_0\Delta u(t+1)+g_1\Delta u(t)}_{\textit{Unknown at time } t} \\
&\quad + \underbrace{\left[G_{d+2}(z^{-1})-g_{d+2,0}-g_{d+2,1}z^{-1}\right]z^2\,\Delta u(t-1)}_{\textit{known at time } t} \\
G_{d+3}(z^{-1})\Delta u(t+d+3-d-1) &= \underbrace{g_0\Delta u(t+2)+g_1\Delta u(t+1)+g_2\Delta u(t)}_{\textit{Unknown at time } t} \\
&\quad + \underbrace{\left[G_{d+3}(z^{-1})-g_{d+3,0}-g_{d+3,1}z^{-1}-g_{d+3,2}z^{-2}\right]z^3\,\Delta u(t-1)}_{\textit{known at time } t}
\end{aligned}
$$

....

$$
\begin{aligned}
G_{d+N}(z^{-1})\Delta u(t+d+N-d-1) &= \underbrace{g_0\Delta u(t+N-1)+g_1\Delta u(t+N-2)+...+g_{N-1}\Delta u(t)}_{\textit{Unknown at time } t} \\
&\quad + \underbrace{\left[G_{d+N}(z^{-1})-g_{d+N,0}-g_{d+N,1}z^{-1}-...-g_{d+N,N-1}z^{-(N-1)}\right]z^N\,\Delta u(t-1)}_{\textit{known at time } t}
\end{aligned}
$$

Thus one of the three terms of *y(t+j)* is dependent on future control actions yet to be determined, another is dependent on the past output values which is known, the rest of them is dependent on future noise signals.

In calculating predictions, the future noise sequences are ignored, thus for *j=d+2* to *j=d+N*:

$$
\begin{aligned}
&\hat{y}(t+d+1)=G_{d+1}\Delta u(t)+F_{d+1}y(t)\\
&\hat{y}(t+d+2)=G_{d+2}\Delta u(t+1)+F_{d+2}y(t)\\
&\cdots\cdots\\
&\cdots\cdots\\
&\hat{y}(t+d+N)=G_{d+N}\Delta u(t+N-1)+F_{d+N}y(t)
\end{aligned}
\tag{2.16}
$$

This can be written as:

$$
Y=\underset{unknown}{GU}+\underset{known}{\Psi} \tag{2.17}
$$

where

$$
Y=\begin{bmatrix}\hat{y}(t+d+1)\\ \hat{y}(t+d+2)\\ \vdots\\ \vdots\\ \hat{y}(t+d+N)\end{bmatrix},\ U=\begin{bmatrix}\Delta u(t)\\ \Delta u(t+1)\\ \vdots\\ \vdots\\ \Delta u(t+N-1)\end{bmatrix},\ G=\begin{bmatrix}g_0 & 0 & \cdots & \cdots & 0\\ g_1 & g_0 & 0 & \cdots & 0\\ g_2 & g_1 & g_0 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ g_{N-1} & g_{N-2} & \cdots & \cdots & g_0\end{bmatrix}
$$

and

$$
\Psi=\begin{bmatrix}\psi(t+d+1) & \psi(t+d+2) & \cdots & \cdots & \psi(t+d+N)\end{bmatrix}^T \tag{2.18}
$$

The components of vector Ψ are known at time t and are given as:

$$
\begin{aligned}
&\psi(t+d+1)=\left[G_{d+1}(z^{-1})-g_{d+1,0}\right]z\,\Delta u(t-1)+F_{d+1}(z^{-1})y(t)\\
&\psi(t+d+2)=\left[G_{d+2}(z^{-1})-g_{d+2,0}-g_{d+2,1}z^{-1}\right]z^2\,\Delta u(t-1)+F_{d+2}(z^1)y(t)\\
&\cdots\cdots\\
&\psi(t+d+N)=\left[G_{d+N}(z^{-1})-g_{d+N,0}-g_{d+N,1}z^{-1}-\ldots-g_{d+N,N-1}z^{-(N-1)}\right]z^N\,\Delta u(t-1)\\
&\qquad\qquad +F_{d+N}(z^{-1})y(t)
\end{aligned}
\tag{2.19}
$$

Let $W=\begin{bmatrix}w(t+d+1) & w(t+d+2) & \cdots & \cdots & w(t+d+N)\end{bmatrix}^T$, the cost function can be written as:

$$\begin{aligned} J &= E\left\{ (GU+\Psi-W)^T (GU+\Psi-W) + \lambda U^T U \right\} \\ &= E\begin{Bmatrix} U^T G^T GU + U^T G^T \Psi - U^T G^T W + \Psi^T GU + \Psi^T \Psi \\ -\Psi^T W - W^T GU - W^T \Psi + W^T W + \lambda U^T U \end{Bmatrix} \end{aligned} \tag{2.20}$$

Now differentiating J with respect to input and equating it to zero:

$$\begin{aligned} \frac{\partial J}{\partial U} &= E\left\{ (GU+\Psi-W)^T (GU+\Psi-W) + \lambda U^T U \right\} \\ &= \frac{\partial}{\partial U}\left[E\begin{Bmatrix} U^T G^T GU + U^T G^T \Psi - U^T G^T w + \Psi^T GU + \Psi^T \Psi - \Psi^T W \\ -W^T GU - W^T \Psi + W^T W + \lambda U^T U \end{Bmatrix} \right] \\ &= 2G^T G + G^T \Psi - G^T W + G^T \Psi - G^T W + 2\lambda U \\ &= 2G^T GU + 2G^T \Psi - 2G^T W + 2\lambda U = 0 \end{aligned} \tag{2.21}$$

$$\Rightarrow \quad U = \left(G^T G + \lambda I\right)^{-1} G^T (W - \Psi)$$

We know that the first element of U is $\Delta u(t) = u(t) - u(t-1)$.

Assume K is the first row of $\left(G^T G + \lambda I\right)^{-1} G^T$ then we could get the control law of MPC:

$$u(t) = u(t-1) + K(W - \Psi) \tag{2.22}$$

After that, assume a system has m-inputs, q-outputs and n_1 states.

If there are more outputs than inputs, it may not be possible control each of the outputs independently with no steady-state error. So the general condition is $q \le m$.

Now considering the effects of noise, represent the system as

$$\begin{cases} x_m(k+1) = A_m x_m(k) + B_m u(k) + B_d \omega(k) \\ y(k) = C_m x_m(k) \end{cases} \tag{2.23}$$

where $\omega(k)$ is the input disturbance and is assumed to be a sequence of integrated white noise.

Now

$$\begin{aligned}&\Delta x_m(k+1)=A_m\Delta x_m(k)+B_m\Delta u(k)+B_d\varepsilon(k)\\&where\quad \varepsilon(k)=\omega(k)-\omega(k-1)\end{aligned}\tag{2.24}$$

Relate $\Delta x_m(k)$ to the output $y(k)$:

$$y(k+1)=C_mA_m\Delta x_m(k)+C_mB_m\Delta u(k)+C_mB_d\varepsilon(k)+y(k)\tag{2.25}$$

Define a new vector as

$$x(k)=\left[\Delta x_m(k)^T\quad y(k)\right]^T,\quad Dimension:(n_1+q)\times 1=n\times 1,\ n=n_1+q$$

The dimensions of different matrix are

$$A_m=n_1\times n_1;\ B_m=n_1\times m,\ C_m=q\times n_1,\ x_m=n_1\times 1,\ \Delta u=m\times 1,\ y=q\times 1$$

Hence the augmented model becomes

$$\begin{cases}\overbrace{\begin{bmatrix}\Delta x_m(k+1)\\ y(k+1)\end{bmatrix}}^{x(k+1)=n\times 1}=\overbrace{\begin{bmatrix}A_m & O_m{}^T\\ C_mA_m & I_{q\times q}\end{bmatrix}}^{A=n\times n}\overbrace{\begin{bmatrix}\Delta x_m(k)\\ y(k)\end{bmatrix}}^{x(k)=n\times 1}+\overbrace{\begin{bmatrix}B_m\\ C_mB_m\end{bmatrix}}^{B=n\times m}\Delta u(k)+\overbrace{\begin{bmatrix}B_d\\ C_mB_d\end{bmatrix}}^{B_\varepsilon=n\times m}\varepsilon(k)\\ y(k)=\overbrace{\begin{bmatrix}O_m & I_{q\times q}\end{bmatrix}}^{C=q\times n}\underbrace{\begin{bmatrix}\Delta x_m(k)\\ y(k)\end{bmatrix}}_{n\times 1},\end{cases}\tag{2.26}$$

$$where\quad O_m=\overbrace{\begin{bmatrix}0 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & 0\end{bmatrix}}^{q\times n_1},n=n_1+q$$

Also we could compute the predicted state variables using future control parameters as:

$$\begin{aligned}x(k_i+1|k_i)&=Ax(k_i)+B\Delta u(k_i)+B_\varepsilon\varepsilon(k_i)\\x(k_i+2|k_i)&=Ax(k_i+1|k_i)+B\Delta u(k_i+1)+B_\varepsilon\varepsilon(k_i+1|k_i)\\&=A^2x(k_i)+AB\Delta u(k_i)+B\Delta u(k_i+1)+AB_\varepsilon\varepsilon(k_i)+B_\varepsilon\varepsilon(k_i+1|k_i)\end{aligned}$$

$$
\begin{aligned}
x\left(k_i+3\middle|k_i\right) &= Ax\left(k_i+2\middle|k_i\right)+B\,\Delta u\left(k_i+2\right)+B_\varepsilon\varepsilon\left(k_i+2\middle|k_i\right)\\
&= A^3x\left(k_i\right)+A^2B\,\Delta u\left(k_i\right)+AB\,\Delta u\left(k_i+1\right)+B\Delta u\left(k_i+2\right)+A^2B_\varepsilon\varepsilon\left(k_i\right)\\
&\quad +AB_\varepsilon\varepsilon\left(k_i+1\middle|k_i\right)+B_\varepsilon\varepsilon\left(k_i+2\middle|k_i\right)
\end{aligned}
$$

……

$$
\begin{aligned}
x\left(k_i+N_p\middle|k_i\right) &= A^{N_p}x\left(k_i\right)+A^{N_p-1}B\,\Delta u\left(k_i\right)+A^{N_p-2}B\,\Delta u\left(k_i+1\right)+....+A^{N_p-N_c}B\Delta u\left(k_i+N_c-1\right)\\
&\quad +A^{N_p-1}B_\varepsilon\varepsilon\left(k_i\right)+A^{N_p-2}B_\varepsilon\varepsilon\left(k_i+1\middle|k_i\right)+.....+B_\varepsilon\varepsilon\left(k_i+N_p-1\middle|k_i\right)
\end{aligned}
$$

With the assumption that $\varepsilon(k)$ is a zero mean white noise sequence, the predicted values of $\varepsilon(k_i+1|k_i)$ at future sample is assumed to be zero. Hence the noise effect to the predicted value is zero.

Now we can compute the predicted output values from predicted state variables as:

$$
\begin{aligned}
&y\left(k_i+1\middle|k_i\right)=CAx\left(k_i\right)+CB\,\Delta u\left(k_i\right)\\
&y\left(k_i+2\middle|k_i\right)=Cx\left(k_i+1\middle|k_i\right)=CA^2x\left(k_i\right)+CAB\,\Delta u\left(k_i\right)+CB\,\Delta u\left(k_i+1\right)\\
&y\left(k_i+3\middle|k_i\right)=Cx\left(k_i+2\middle|k_i\right)=CA^3x\left(k_i\right)+CA^2B\,\Delta u\left(k_i\right)+CAB\,\Delta u\left(k_i+1\right)+CB\Delta u\left(k_i+2\right)\\
&\ldots\ldots\\
&y\left(k_i+N_p\middle|k_i\right)=CA^{N_p}x\left(k_i\right)+CA^{N_p-1}B\,\Delta u\left(k_i\right)+CA^{N_p-2}B\,\Delta u\left(k_i+1\right)+....+CA^{N_p-N_c}B\Delta u\left(k_i+N_c-1\right)
\end{aligned}
$$

Thus we can write:

$$Y=F\,x\left(k_i\right)+\varphi\,\Delta U$$

$$
where\ \ F=\overbrace{\begin{bmatrix} CA\ \ (q\times n)\\ CA^2\\ CA^3\\ ....\\ CA^{N_p}\end{bmatrix}}^{N_pq\times n};\ \varphi=\overbrace{\begin{bmatrix} CB(q\times m) & 0 & 0 & .... & 0\\ CAB & CB & 0 & .... & 0\\ CA^2B & CAB & CB & .... & 0\\ ... & .... & .... & .... & ....\\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & .... & CA^{N_p-N_c}B\end{bmatrix}}^{N_pq\times N_cm}
$$

$$Y=N_pq\times 1,\ \ x=n\times 1,\ F=N_pq\times n,\ \ \varphi=N_pq\times N_cm,\ \Delta U=N_cm\times 1$$

Then

$$
\begin{aligned}
\Delta U &= \left(\varphi^T\varphi+Q\right)^{-1}\varphi^T\left(\overline{R_s}\,r\left(k_i\right)-Fx\left(k_i\right)\right)\\
&= \left(\varphi^T\varphi+Q\right)^{-1}\varphi^T\,\overline{R_s}\,r\left(k_i\right)-\left(\varphi^T\varphi+Q\right)^{-1}\varphi^TFx\left(k_i\right)
\end{aligned}
\tag{2.27}
$$

Again, we could find the control law applying receding horizon principle

$$\Delta u(k_i) = \overbrace{\left[I_m \quad O_m \quad .. \quad .. \quad O_m\right]}^{m\times N_c}\left[\left(\varphi^T\varphi+Q\right)^{-1}\right]\left[\varphi^T \overline{R_s}\, r(k_i)-\varphi^T F x(k_i)\right]$$
$$= K_y r(k_i) - K_{mpc} x(k_i) \tag{2.28}$$

With dimentions: $K_y = m\times q,\ K_{mpc} = m\times n,\ \Delta u = m\times 1$

2.2.3 Simulation Result

The block diagram of an isolated thermal power system is shown in figure 2.1 with the parameters in table 2.1.

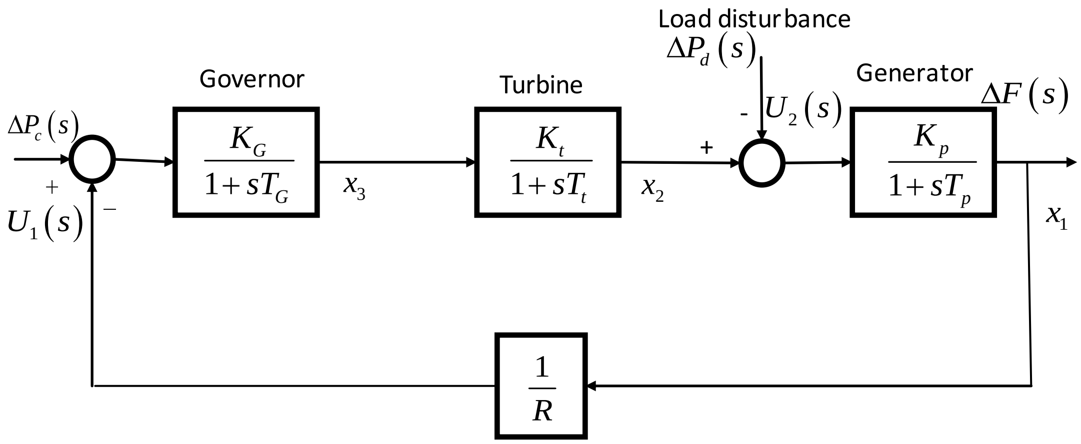

Figure 2.1. Block diagram of an isolated thermal power system

T_p	K_p	T_G	K_G	T_t	K_t	R
20	120	0.08	0.6	0.3	1.0	2.4

Table 2.1. Parameters of an isolated thermal power system

Firstly find out the state equations describing the system:

$$\begin{cases} \dot{x}_1 = -\dfrac{1}{T_p}x_1 + \dfrac{K_p}{T_p}x_2 - \dfrac{K_p}{T_p}\Delta P_d = -\dfrac{1}{T_p}x_1 + \dfrac{K_p}{T_p}x_2 - \dfrac{K_p}{T_p}u_2 \\ \dot{x}_2 = -\dfrac{1}{T_t}x_2 + \dfrac{K_t}{T_t}x_3 \\ \dot{x}_3 = -\dfrac{K_g}{T_g R}x_1 - \dfrac{1}{T_g}x_3 + \dfrac{K_g}{T_g}u_1 \end{cases} \tag{2.29}$$

Where $u_1 = \Delta P_c, u_2 = \Delta P_d$

Thus

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -\frac{1}{T_p} & \frac{K_p}{T_p} & 0 \\ 0 & -\frac{1}{T_t} & \frac{K_t}{T_t} \\ -\frac{K_g}{T_g R} & 0 & -\frac{1}{T_g} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{K_g}{T_g} \end{bmatrix} u_1 + \begin{bmatrix} -\frac{K_p}{T_p} \\ 0 \\ 0 \end{bmatrix} u_2 \tag{2.30}$$

Transfer it from continuous time to discrete time at a sample time 0.01s in Matlab. We could get the discrete time system state equation:

$$\begin{cases} x_m(k+1) = \begin{bmatrix} 0.9995 & 0.0590 & 0.0009 \\ -0.0008 & 0.9672 & 0.0308 \\ -0.0489 & -0.0015 & 0.8825 \end{bmatrix} x_m(k) + \begin{bmatrix} 0 \\ 0.002 \\ 0.1175 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_m(k) \end{cases} \tag{2.31}$$

Set $q_w=50$, $N_p=10$, $N_c=2$, than calculate the control law of the discrete system, we could get:

$$K_y = 0.0024$$
$$K_{mpc} = \begin{bmatrix} 0.0024 & 0.0001 & 0 & 0.0024 \end{bmatrix}$$

Thus we could simulate the close loop system and compare its step response to the original system shown in figure 2.2:

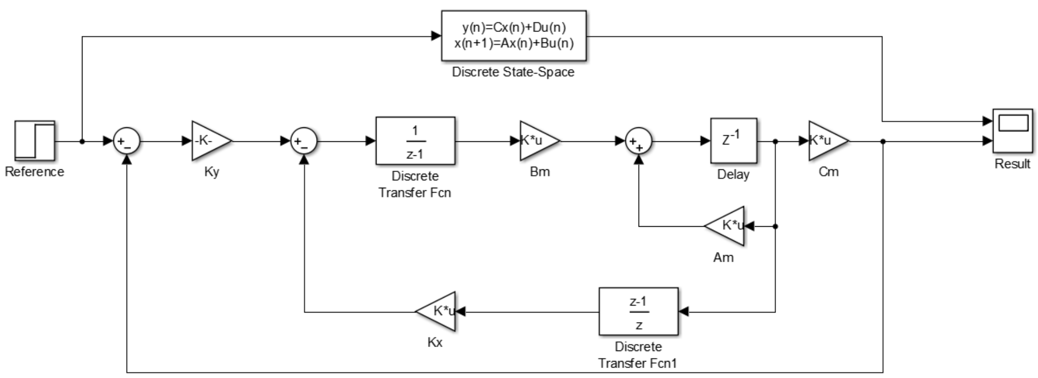

Figure 2.2. Step response for MPC control system and the original system

The simulation results for steady state, which means we assume the load disturbance is 0, are shown in figure 2.3:

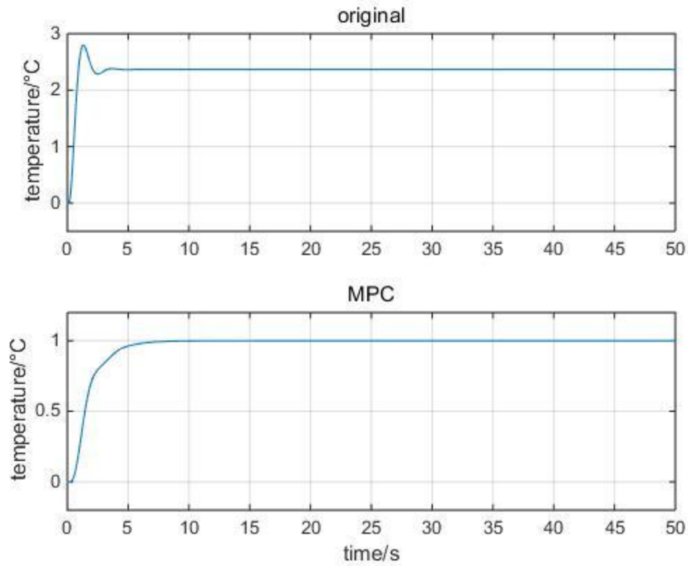

Figure 2.3. System performance with noise

We could see that the system response could accurately track the reference signal (input) after adopting MPC controller.

Then, add the load disturbance into the system to test if the system is stable in steady state. Figure 2.4 shows the system responses in steady state while the load disturbance changes.

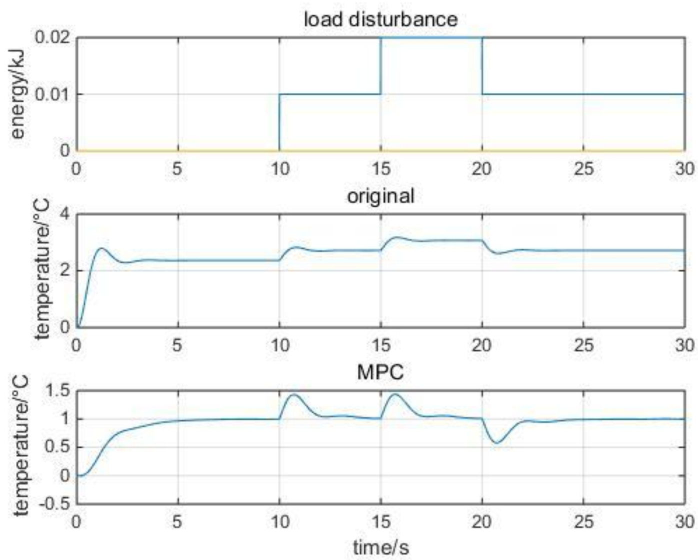

Figure 2.4. System response with load disturbance

We could see that the system could rapidly get back to steady state with MPC controller, thus it is robust.

2.3 Bit-Stream based discrete time MPC Control

2.3.1 The concept of Bit-Stream and Δ-Σ Modulator

Bit-stream based technique can convert either an analog or a multi-bits digital signal into a single-bit output. Compared to traditional multi-bits processing, bit-stream system use a single channel instead of multiple channel between subsystems. Thus it consumes less hardware resources.[5]

The input analogue/multi-bits signal is encoded into a bit-stream signal by a Delta-Sigma modulator at high sampling frequencies. Figure 2.5 shows the model of a first order Delta-Sigma modular.

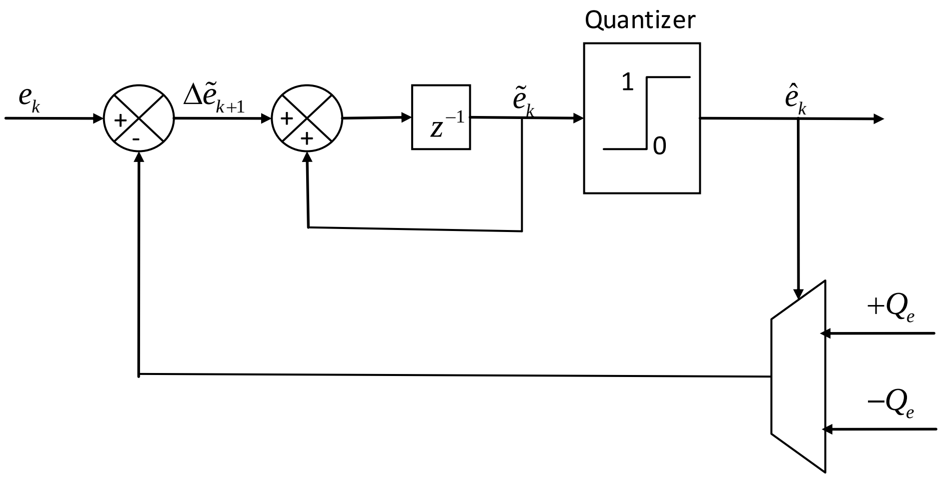

Figure 2.5. A fist order Delta-Sigma modulator

2.3.2 Bit-Stream Stability analysis

Assume the closed loop system transfer function can be expressed as:

$$H(s) = \frac{G_c(s)G_p(s)}{1+G_c(s)G_p(s)} \tag{2.32}$$

$G_c(s)$ is the controller model of the system and $G_p(s)$ is the system plant model.

If the controller output stabilizes the system perfectly, then all the poles of the transfer function $H(s)$ should be negative and real. In order to convert the control loop to a bit-stream based model, a Delta-Sigma modulator is adopted to convert the error signal e into a switch signal $\hat{e}$. The bit-stream based control system is shown in figure 2.6.

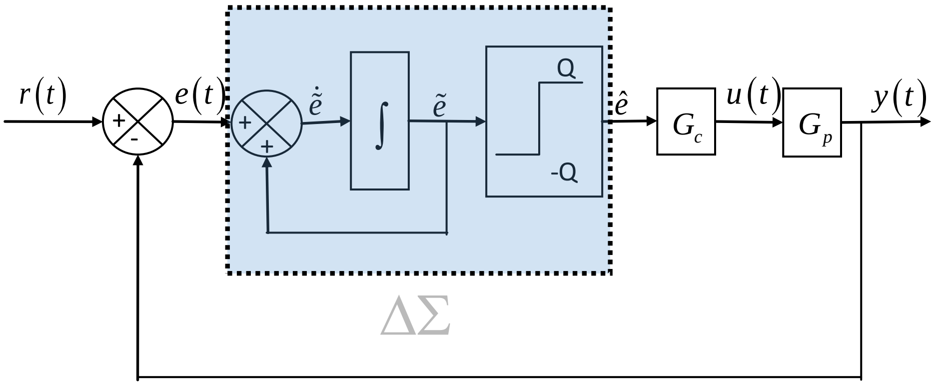

Figure 2.6. Bit-stream based control system.

It is very important to investigate the stability of the bit-stream based control system as the nonlinearity in Delta-Sigma modulator might affect the system stability.

Firstly find out the coupling between the system and Delta-Sigma modulator:

$$\frac{y}{\hat{e}} = G_c(s)G_p(s) \tag{2.33}$$

$$e = r - y \tag{2.34}$$

$$\begin{aligned}\hat{e} &= Q\,\mathrm{sgn}\left(\int_0^t e - \hat{e}\; dt\right) \\ &= Q\,\mathrm{sgn}(\tilde{e})\end{aligned} \tag{2.35}$$

where the sign function is known as

$$\operatorname{sgn}(\tilde{e}) = \begin{cases} +1 & if \ \hat{e} \geq 0 \\ -1 & if \ \hat{e} < 0 \end{cases} \tag{2.36}$$

We could figure out that based on the contact between the system and Delta-Sigma modulator, Q should be chosen after considering the whole system.

Sliding mode analysis suggests a Lyapunov function:

$$V = \frac{1}{2}\tilde{e}^2 \tag{2.37}$$

The derivative of the function would be:

$$\begin{aligned} \dot{V} &= \tilde{e}\dot{\tilde{e}} < 0 \\ &= \tilde{e}\left(e - Q\operatorname{sgn}(\tilde{e})\right) < 0 \end{aligned} \tag{2.38}$$

From it we could find that the error e must be bounded by the Delta-Sigma modulator output Q in order to ensure the existence of the sliding motion on $\tilde{e} = 0, \dot{\tilde{e}} = 0$. This could be written as:

$$\|e\|_\infty \leq Q \tag{2.39}$$

Hence, $\dot{V}$ is negative always. From the point of view of sliding mode, the Delta-Sigma modulator output which is a switched signal is identical to modulator the input [6]. Assume the ideal sliding mode exists, then $\tilde{e} = 0, \dot{\tilde{e}} = 0$ are forced to zero at an infinite sampling frequency, ensure that

$$e \equiv \hat{e} \tag{2.40}$$

However, the ideal sliding mode is impossible to achieve. Hence, we should consider the real sliding mode at a finite sampling frequency instead of ideal sliding mode assumption.

From figure 2, we could define:

$$\Delta\tilde{e}_{k+1} = \tilde{e}_{k+1} - \tilde{e}_k \tag{2.41}$$

Thus the Lyapunov function of the discrete-time Delta-Sigma modulator is:

$$V = \frac{1}{2}\tilde{e}_k^2 \tag{2.42}$$

One of the condition of stability requires the difference of the Lyapunov function should meet:

$$\begin{aligned}\Delta V &= \tilde{e}_k \Delta\tilde{e}_{k+1} < 0 \\ &= \tilde{e}\left(e_k - Q\,\mathrm{sgn}\left(\tilde{e}_k\right)\right) < 0 \\ &= \tilde{e}_k\left(\tilde{e}_{k+1} - \tilde{e}_k\right) < 0\end{aligned} \tag{2.43}$$

Furthermore, this equation can be written as:

$$\begin{cases}\tilde{e}_{k+1} < \tilde{e}_k & \forall\tilde{e}_k > 0 \\ \tilde{e}_{k+1} > \tilde{e}_k & \forall\tilde{e}_k < 0\end{cases} \tag{2.44}$$

So $\tilde{e}_k$ points toward zero each sampling time. Because of the imperfection of the switch elements, the equation $\tilde{e} = 0$ is replaced by:

$$\left\|\tilde{e}_k\right\|_\infty \le 2Q \tag{2.45}$$

Considering the close loop transfer function of the system and the reference signal r, Q is obtained as the follow equation [35]:

$$\left\|e_\infty\right\| \le \left\|\frac{1}{1+G_cG_p}\right\|_1 \left\|r\right\|_\infty \le Q \tag{2.46}$$

Where,

$$\left\|X\right\|_i \square \left(\frac{1}{2\pi}\int_{-\pi}^{\pi}\left|X\left(e^{j\omega}\right)\right|^i d\omega\right)^{\frac{1}{i}} \tag{2.47}$$

This means that Q must be greater than the input of Delta-Sigma modulator to keep the system stable.

2.3.3 MATLAT toolbox for Δ-Σ Modulator

In order to widely apply the Delta-Sigma modulator on different types of control methods and systems simulation, a MATLAT toolbox which includes a BS-converter and a BS-inverter is developed.

a) BS-converter

Firstly, a block model of BS-converter is developed.

Below in figure 2.7 shows the block for the Delta-Sigma modulator, it converts the input signal to a digital signal.

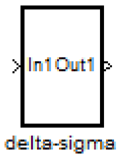

Figure 2.7. A block model for Delta-Sigma modulator in MATLAB

The internal structure of the block is shown in figure 2.8.

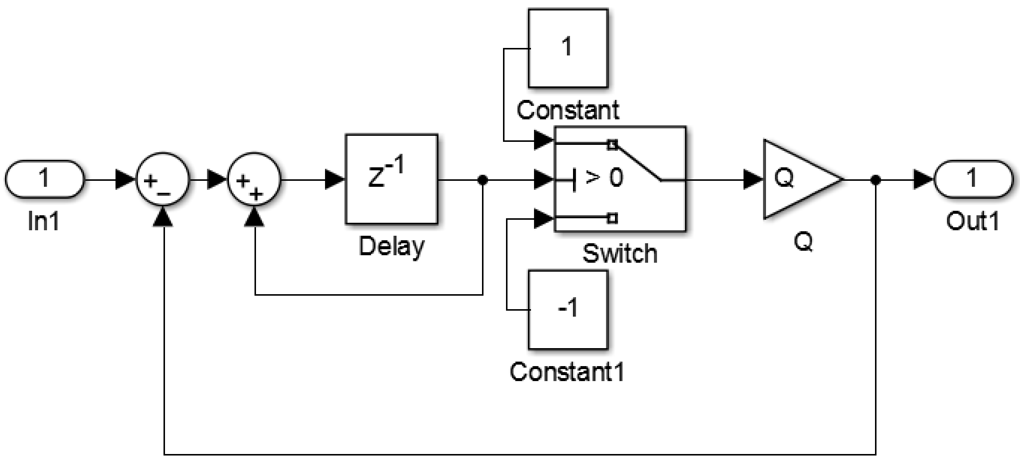

Figure 2.8. The internal structure of the BS-converter

where Q is the parameter of the modulator block.

Q must be greater than or equal to the modulator input to ensure the stability of Delta-sigma modulator.

b) BS-inverter

Then, it is necessary to check if the output signal represents the true input signal. So a block model of BS-inverter is also developed. The digital signal can be assumed as the input signal with high frequency noises, so the BS-inverter is actually a low pass discrete filter. [7]

The inverter block is shown in figure 2.9 and its internal structure is shown as figure 2.10.

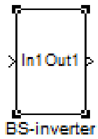

Figure 2.9. A block model for BS-inverter in MATLAB

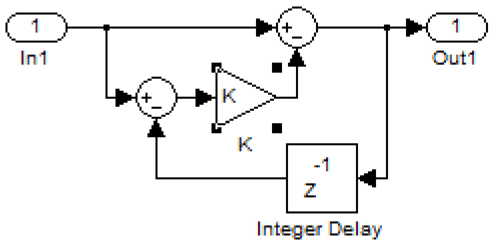

Figure 2.10. The internal structure of the BS-inverter

K is the parameter of the inverter and it is obtained from:

$$K = \frac{f_s/2\pi f_c}{1 + f_s/2\pi f_c} \tag{2.48}$$

where f_s is the sampling frequency and f_c is the cutoff frequency.

Consider a sine wave input signal shown in figure 2.11 and check if the inverter works. The simulation result is shown in figure 2.12. It is observed that although some attenuation occurs because of the filter, it still restores the input signal.

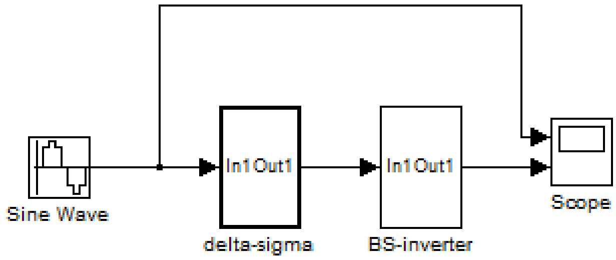

Figure 2.11. The simulation of the inverter in MATLAB

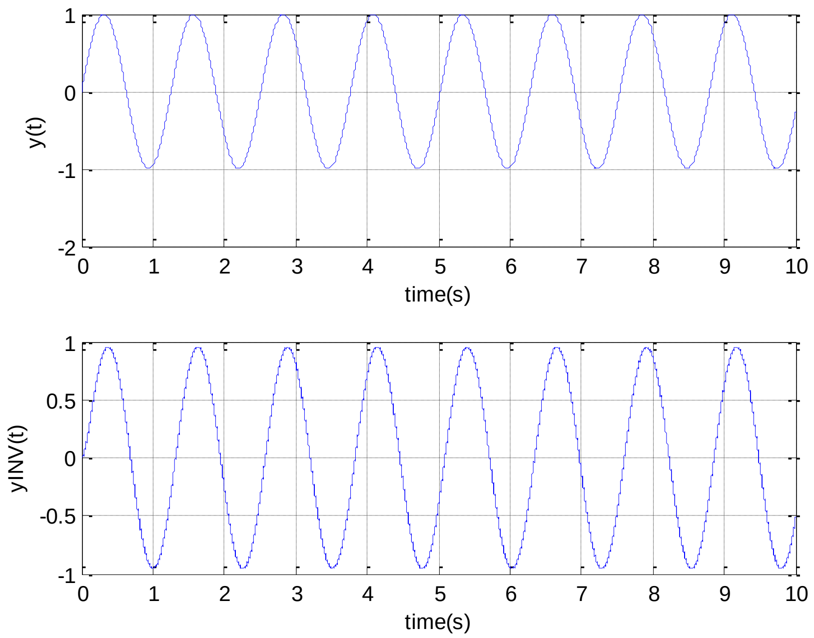

Figure 2.12. Original input signal (top) and the output signal though BS-inverter

2.3.4 Bit-Stream Based discrete time MPC Control for Linear System

Considering the isolated thermal power system discussed in 2.2.3. The MPC control law is not changed.

As mentioned in 2.3.2, in order to convert the control loop to a bit-stream based model, a Delta-Sigma modulator is adopted to convert the error signal e into a switch signal $\hat{e}$ which is shown in figure 2.13.

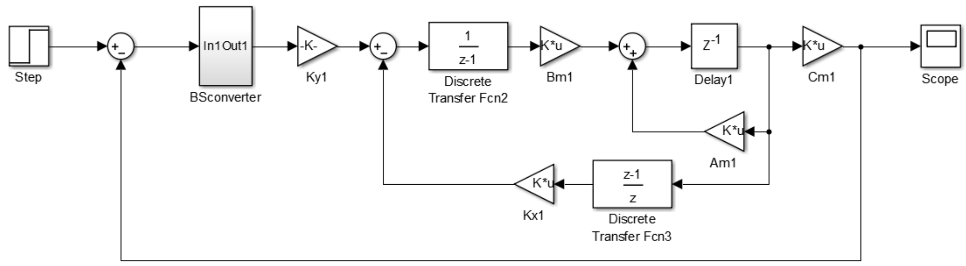

Figure 2.13. Bit-stream based MPC control system.

Figure 2.14 shows the steady state and load disturbance response of the bit-stream based MPC control system.

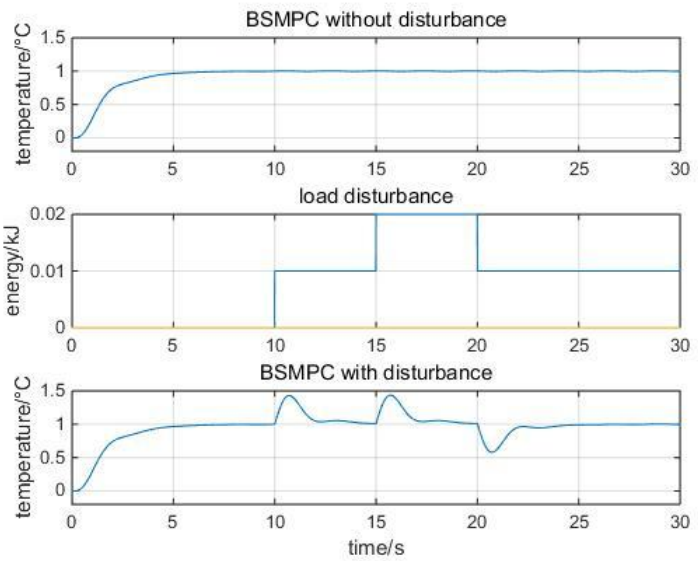

Figure 2.14. Steady state and Load disturbance response of Bit-stream MPC control system.

We could see that the system keeps its stability and robustness after adopting the Delta-Sigma modulator.

2.4 Summary

The model predictive controller in discrete time domain has several advantages compared with conventional controllers. Also after adding Delta-Sigma modulator the system performance remains the same.

However, comparing the system response of MPC and bit-stream based MPC in figure 2.15, we could find that the bit-stream based MPC has more ripples thus has more noise than MPC.

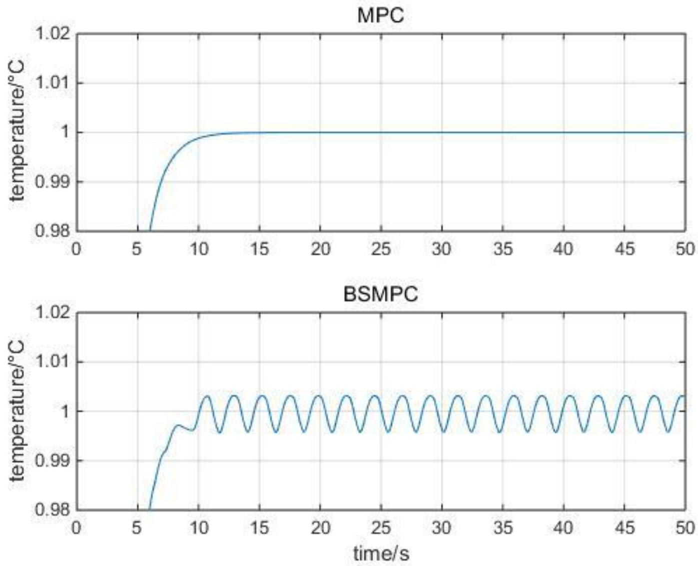

Figure 2.15. Ripples of MPC and Bit-stream MPC control system.

This phenomenon happens because bit-stream technology is more suitable for continuous time system because the Delta-Sigma modulator contains a switch with specific operating frequency and that would lead to some noise. In discrete time system, if the switch frequency is different from the system sampling frequency, the system would become unstable. Therefore, the switch frequency of the modulator should be the same as the system sampling frequency, in this case *100Hz,* and thus lower frequency of switch operation would lead to a higher noise.

The next chapter would investigate the performance of Delta-Sigma modulator in continuous time domain which may reduce the ripples in a higher frequency.

Chapter 3

Bit-Stream based Continuous Time Model Predictive Control for Linear System

3.1 Introduction

In chapter 2, the performance of MPC and bit-stream based MPC were demonstrated on specific system. However, comparing with traditional MPC, bit-stream based MPC has some flaws in controlling the discrete time systems as the switching frequency should be the same as the system sampling frequency.

The rest of the present study therefore focus on continuous time systems. The research on continuous time MPC (CMPC) design is based on discrete time approach and can overcome some of the disadvantages of discrete time MPC. This control algorithm was proposed by Demircioglu et al. in 1991 [8].

This chapter firstly introduce the linear CMPC algorithm based on state space model in section 3.2. Then a bit-stream based CMPC is brought up in this condition. After all of this, both CMPC and bit-stream based CMPC are applied on a specific system and simulations are obtained in order to investigate the performance of these controllers.

3.2 CMPC for linear system

3.2.1 Control Law

Consider a system which can be expressed as:

$$A(s)Y(s) = B(s)U(s) + C(s)X(s) \tag{3.1}$$

where $A(s)$, $B(s)$ and $C(s)$ are the polynomials in the Laplace operator s. $Y(s)$, $U(s)$ and $X(s)$ are the system output, control input and disturbance input.

Firstly we predict the future output in a time period of T in time domain:

$$\hat{y}(t+T) \tag{3.2}$$

This predictor can be expressed by a Taylor series expansion, thus:

$$\hat{y}(t+T) = y(t) + \sum_{k=0}^{N_y} y_k(t)\frac{T^k}{k!} \tag{3.3}$$

where

$$y_k(t) = \frac{d^k \hat{y}(t)}{dt^k}$$

and N_y is the order of the predictor.

As discussed in [8], generally speaking, large future time T corresponding to a high order of predictor N_y in order to achieve a good prediction.

In Laplace domain, the kth derivative corresponds to orders of s. Thus the system output in Laplace domain can be written as:

$$Y_k(S) = s^k Y(s) = \frac{s^k B(s)}{A(s)}U(s) + \frac{s^k C(s)}{A(s)}X(s) \tag{3.4}$$

where *X(s)* is the disturbance.

Assume that the disturbance term can be decomposed as:

$$\frac{s^k C(s)}{A(s)} = E_k(s) + \frac{F_k(s)}{A(s)} \tag{3.5}$$

Then the output equation is derived as,

$$Y_k(s) = Y_k^*(s) + E_k^*(s) \tag{3.6}$$

where

$$Y_k^*(s) = \frac{s^k B(s)}{A(s)} U(s) + \frac{F_k(s)}{A(s)} \tag{3.7}$$

and

$$E_k^*(s) = E_k X(s) \tag{3.8}$$

Substituting (3.1) and (3.5) into (3.7) we could get,

$$Y_k^*(s) = \frac{E_k B}{C} U(s) + \frac{F_k}{C} Y(s) \tag{3.9}$$

The term $\frac{E_k B}{C}$ is not a proper transfer function for $k > d$, where d is the relative order of the system. This term can be written as a strictly proper part and a remainder polynomial.

$$\frac{E_k B}{C} = \frac{G_k}{C} + H_k \tag{3.10}$$

So, (3.9) can be written as:

$$Y_k^*(s) = H_k U(s) + \underbrace{\frac{G_k}{C} U(s) + \frac{F_k}{C} Y(s)}_{realizable} \tag{3.11}$$

The equation has both realizable and unrealizable part. We define the realizable part as:

$$Y_k^0(s) = \frac{G_k}{C} U(s) + \frac{F_k}{C} Y(s) \tag{3.12}$$

Thus,

$$Y_k^*(s) = H_k U(s) + Y_k^0(s) \tag{3.13}$$

In the time domain, (3.13) is expressed as:

$$y_k^*(t) = h_k u + y_k^0(t) \tag{3.14}$$

where h_k is a row vector, it contains the coefficients of H_k polynomial. The elements of h_k are the system Markov parameters.

u is a column vector which contains the derivatives of the input:

$$u = \begin{bmatrix} u(t) & u^{[1]}(t) & \cdots & u^{[k-p]}(t) \end{bmatrix}^T \tag{3.15}$$

where

$$u^{[i]} = \frac{d^i u(t)}{dt^i}$$

As stated in [8], the prediction is established by the Taylor series expansion of the system output. Thus we could express the T time predictor in matrix form as:

$$y^*(t+T) = T_{N_y} H u + T_{N_y} Y^0 \tag{3.16}$$

where

$$T_{N_y} = \begin{bmatrix} 1 & T & \frac{T^2}{2!} & \cdots & \frac{T^{N_y}}{N_y!} \end{bmatrix}$$

$$Y^0 = \begin{bmatrix} y(t) & y_1^0(t) & \cdots & y_{N_y}^0(t) \end{bmatrix}^T$$

and H is the Markov parameter matrix of the polynomials H_k which dimension $\left(N_y+1\right)\times\left(N_y-d+1\right)$.

When $d=1$, H is written as:

$$H = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ h_1 & 0 & 0 & \cdots & 0 \\ h_2 & h_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{N_y} & h_{N_y-1} & h_{N_y-2} & \cdots & h_1 \end{bmatrix} \tag{3.17}$$

The aim of CMPC is to control the predicted future output as close as possible to the future set point. This means that the future set point has to be known.

However, in many cases, the exact future set point is unknown. In that case, we usually assume a constant set point *w* into the future. If we try to control the predicted future to match this constant set point, the output would follow the future set point very fast but also an overshoot may occur. To avoid the overshoots, we could consider a smooth approximation from output to the constant set point as shown in figure 3.1.

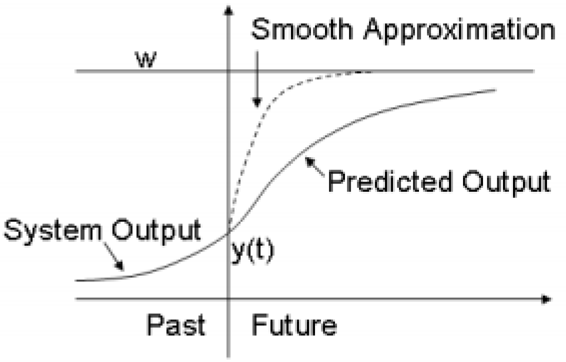

Figure 3.1 the Smooth Approximation

The reference trajectory *R(t,T)* will be taken as the output of the rational transfer function with numerator R_n and denominator R_d. And

$$R(t,s)=\frac{R_n(s)}{R_d(s)}\frac{\left[w(t)-y(t)\right]}{s} \tag{3.18}$$

The Laplace operator s denotes the Laplace transform with respect to future variable T. The Taylor series expansion of R_n/R_d can be expressed as:

$$\frac{R_n}{R_d}=\sum_{i=0}^{N_y}r_i s^{-1} \tag{3.19}$$

where r_i is the Markov parameter of R_n/R_d. Substituting (3.19) to (3.18) and taking the inverse Laplace transform, the smooth approximation of reference trajectory in matrix form is:

$$R^*(t+T) = T_{N_y} w \tag{3.20}$$

where

$$\begin{aligned} w &= R[w(t) - y(t)] \\ R &= \begin{bmatrix} r_0 & r_1 & \cdots & r_{N_y} \end{bmatrix}^T \end{aligned} \tag{3.21}$$

How to choose the reference trajectory, a constant future set point or a smooth approximation, would depend on the requirement of the system. A constant future set point would lead to a fast system respond with overshoot and a smooth approximation would lead to a slow system response with no overshoot.

In discrete time MPC, the other key parameter of the controller design is control horizon N_u. And in continuous time, it is called control order and is defined by:

$$For\ k > N_u \qquad u^{[k]}(t) = 0$$

where $u^{[k]}(t) = \dfrac{d^k u(t)}{dt^k}$

By introducing the control order, the dimension of vector u is reduced to $(N_u+1)X1$ and Markov parameter matrix is reduced to $(N_y+1)X(N_u+1)$.

The control law is based on minimizing at the instant t a quadratic cost function of the output tracking error and the control values. And in CMPC, the cost function is defined as:

$$
\begin{aligned}
J = &\int_{T_1}^{T_2}\left[y^*(t+T)-R^*(t+T)\right]^T\left[y^*(t+T)-R^*(t+T)\right]dT \\
&+\int_0^{T_2-T_1}u^*(t+T)^T Q u^*(t+T)dT
\end{aligned} \tag{3.22}
$$

where T_1 is the minimum prediction horizon, T_2 is the maximum prediction horizon, Q is the control weighting and the truncated Taylor series expansion form of the predicted future input is:

$$u^*(t+T)=T_{N_u}u \tag{3.23}$$

$$T_{N_u}=\begin{bmatrix}1 & T & \dfrac{T^2}{2!} & \cdots & \dfrac{T^{N_u}}{N_u!}\end{bmatrix} \tag{3.24}$$

$$u=\begin{bmatrix}u(t) & u^{[1]}(t) & \cdots & u^{[N_u]}(t)\end{bmatrix}^T \tag{3.25}$$

Thus we could minimization J to get a result:

$$u=K(w-Y^0) \tag{3.26}$$

where

$$K=(H^T T_y H+QT_u)^{-1}H^T T_y \tag{3.27}$$

$$T_y=\int_{T_1}^{T_2}T_{N_y}^T T_{N_y}dT \tag{3.28}$$

$$T_u=\int_0^{T_2-T_1}T_{N_u}^T T_{N_u}dT \tag{3.29}$$

By taking the first row of K be k, the control law of CMPC is expressed as,

$$u(t)=k(w-Y^0) \tag{3.30}$$

When the system is linear, the state space approach and transfer function approach of design CMPC are almost the same. However, for nonlinear systems, the design of CMPC needs to be done using state space approach. Therefore, it is necessary to design linear system using state space approach for the further use of nonlinear system design.

Consider a state space system with the form:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{3.31}$$

where *x, y, u* are state variable, system output and system input (control input). Because the disturbance doesn't have an effect on the predictive output and also for simplicity, the disturbance part is omitted.

The future development of a continuous signal can be obtained by taking derivatives. Thus we could repeat differentiation of *y* and get:

$$Y_{N_y}(t) = O_{N_y} x(t) + H_{N_y,N_y} U_{N_u}(t) \tag{3.32}$$

where N_y is the order of highest derivative of output *y*

N_u is the order of highest derivative of output *u*.

The output derivatives vector is written as

$$Y_{N_y}(t) = \begin{bmatrix} y & y^{[1]} & y^{[2]} & \cdots & y^{[N_y]} \end{bmatrix}^T \tag{3.33}$$

The input derivative vectors is written as

$$U_{N_u}(t) = \begin{bmatrix} u & u^{[1]} & u^{[2]} & \cdots & u^{[N_u]} \end{bmatrix}^T \tag{3.34}$$

O_{N_y} is the extended observability matrix with $\left(N_y+1\right)\times n$ dimension. n is the dimension of system state variables.

$$O_{N_y} = \begin{bmatrix} C & CA & CA^2 & \cdots & CA^{N_y} \end{bmatrix}^T \tag{3.35}$$

H_{N_y,N_y} is the Markov matrix:

$$H_{N_y,N_y} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{N_y-1}B & CA^{N_y-2}B & CA^{N_y-3}B & \cdots & CA^{N_y-N_u-1}B \end{bmatrix} \tag{3.36}$$

The predicted output *Y(t)* is defined as

$$\hat{Y}_{N_y}(t) = O_{N_y}\hat{x}(t) + H_{N_y,N_y}U_{N_u}(t) \tag{3.37}$$

The control weighting is set to zero because it is not considered as important as prediction order and control order [9]. It is assumed that the reference signal $w(t,\tau)$ has a Taylor series expansion. We use τ to indicate time, then the reference signal can be expressed as:

$$w(t,\tau) = T(\tau)W(t) \tag{3.38}$$

where

$$T(\tau) = \begin{bmatrix} 1 & \tau & \dfrac{\tau^2}{2!} & \cdots & \dfrac{\tau^{N_y}}{N_y!} \end{bmatrix}$$

$$W(t) = R_0 y(t) + R(w(t) - y(t))$$

The R term is a column vector containing the Markov parameters of a reference dynamic system [8]. The first element of R_0 is 1 and rest are 0. For simplicity, set R equals to R_0.

Then

$$W(t) = R_0 w(t) \tag{3.39}$$

Consider the following cost function:

$$J = \int_{\tau_1}^{\tau_2} \left[y^*(t+\tau) - W^*(t+\tau) \right]^T \left[y^*(t+\tau) - W^*(t+\tau) \right] d\tau \tag{3.40}$$

Parameterize the future time variable T into the scale $\alpha \in (0,1)$, get

$$\tau = \tau_1 + \alpha(\tau_2 - \tau_1) \tag{3.41}$$

Then substituting (3.37) and (3.39) into (3.40), we could write the cost function as

$$
\begin{aligned}
J &= \int_0^1 \left[O_{N_y} x + H_{N_y,N_y} U_{N_u} - W \right]^T T^T(\tau) T(\tau) \left[O_{N_y} x + H_{N_y,N_y} U_{N_u} - W \right] d\alpha \\
&= \left[O_{N_y} x + H_{N_y,N_y} U_{N_u} - W \right]^T \bar{T}(\tau_1, \tau_2) \left[O_{N_y} x + H_{N_y,N_y} U_{N_u} - W \right]
\end{aligned} \tag{3.42}
$$

where

$$
\bar{T}(\tau_1, \tau_2) = \int_0^1 T^T(\tau) T(\tau) d\alpha
$$

The elements of matrix $\bar{T}(\tau_1, \tau_2)$ can be calculated from the equation

$$
\bar{T}_{ij}(\tau_1, \tau_2) = \frac{\tau_2^{i+j-1} - \tau_1^{i+j-1}}{(i-1)!(j-1)!(i+j-1)} \tag{3.43}
$$

then we can obtain the matrix $\bar{T}(\tau_1, \tau_2)$.

Set the result of first derivative of the cost function to zero we could obtain U_{N_u}. The first element of U_{N_u} is used as the control input.

$$
U_{N_u} = \left[H_{N_y,N_y}^T \bar{T}(\tau_1, \tau_2) H_{N_y,N_y} \right]^{-1} \left[H_{N_y,N_y}^T \bar{T}(\tau_1, \tau_2) \right] \left(W - O_{N_y} \hat{x}(t) \right) \tag{3.44}
$$

3.2.2 Simulation Result

Consider the isolated thermal power system discussed in 2.2.3. This time we don't need to transfer the system from continuous time to discrete time.

Thus, figure out the system function as

$$
\begin{cases}
\dot{x}(t) = \begin{bmatrix} -0.05 & 6 & 0 \\ 0 & -3.3333 & 3.3333 \\ -3.125 & 0 & -12.5 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 7.5 \end{bmatrix} u(t) \\
y(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t)
\end{cases} \tag{3.45}
$$

Then choose the parameters of MPC controller

$$
N_y = 4, N_u = 1, \tau_1 = 0, \tau_2 = 1,
$$

Therefore we could calculate the control law of the control input:

$$
u(t) = K_w \left[w(t) - K_y y(t) \right] - K_x x(t) \tag{3.46}
$$

In this case

$$
\begin{aligned}
K_x &= [-0.4277 \quad 0.7126 \quad -1.0511] \\
K_y &= 1 \\
K_w &= 0.4480
\end{aligned}
$$

Then we could build the system model in MATLAB/Simulink in figure 3.2 and obtain the simulation results of the original system and the CMPC control system.

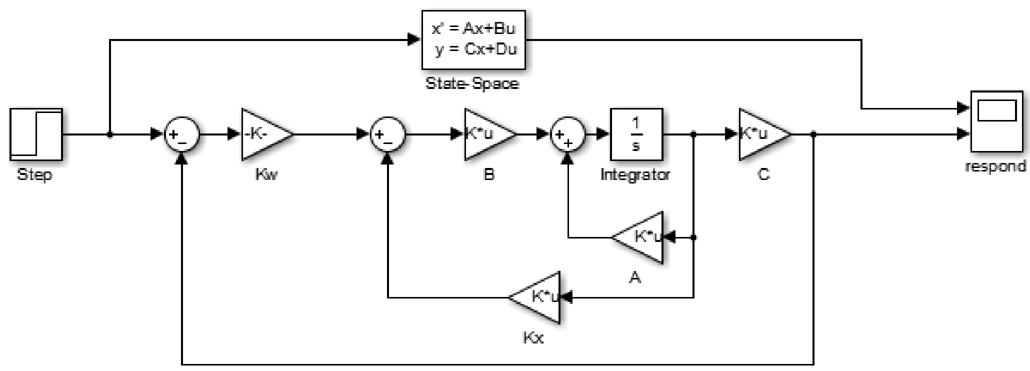

Figure 3.2. System model for the original plant system and the CMPC control system..

Figure 3.3 shows the simulation results.

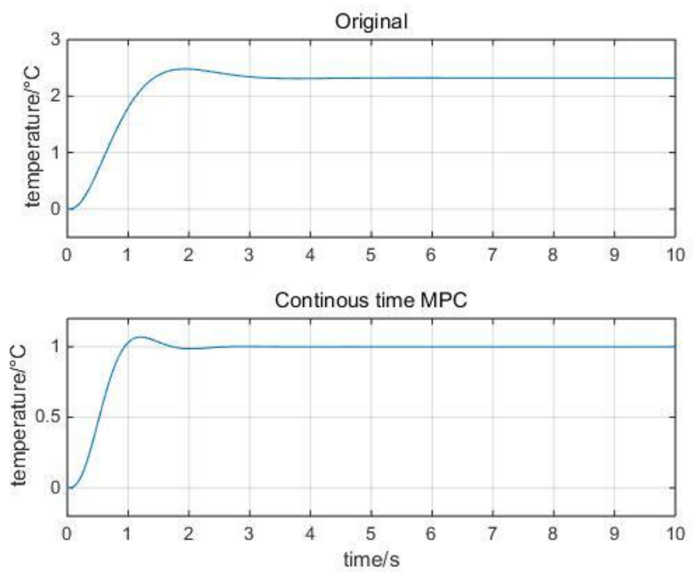

Figure 3.3. System step response of the original plant system and the CMPC control system.

We could find out that the CMPC control system has a faster reaction and also it tracks the reference signal with higher accuracy.

3.3 Bit-Stream Based CMPC Control for Linear System

Considering the isolated thermal power system. The MPC control law is not changed.

In order to convert the control loop to a bit-stream based model, a Delta-Sigma modulator is adopted between the controller and the system to reduce the communication channel which is shown in figure 3.4.

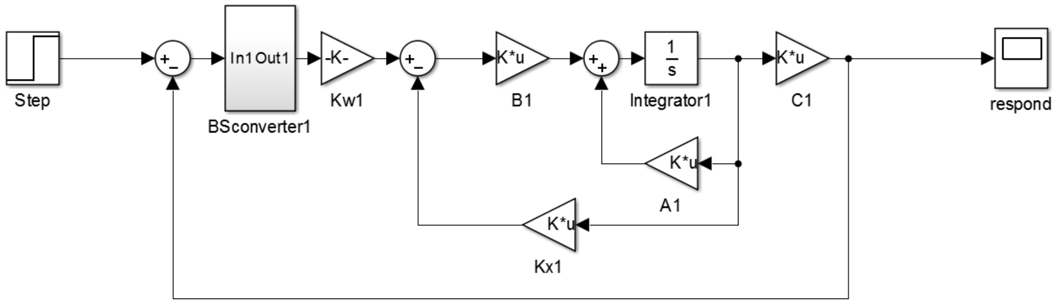

Figure 3.4. Bit-stream based CMPC control system.

Figure 3.5 shows the step response of the bit-stream based CMPC control system.

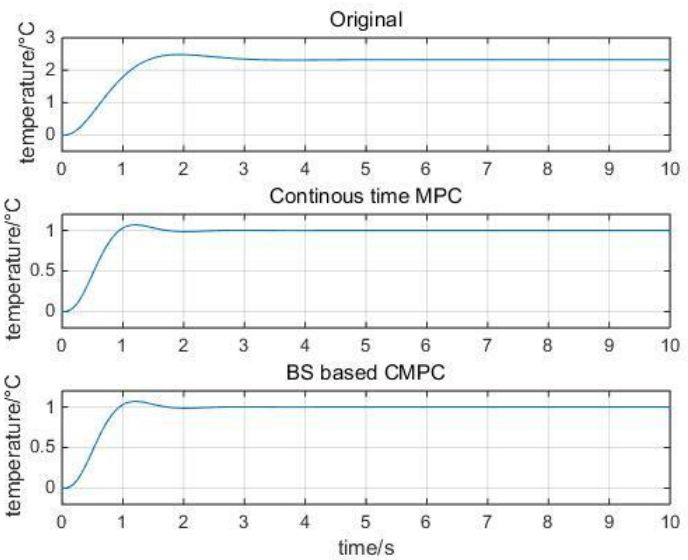

Figure 3.5. Step response of the bit-stream based CMPC control system

Compare the results between the plant system, CMPC control system and bit-stream based CMPC control system, we could find that the system keeps its stability and performance after adopting the Delta-Sigma modulator and both CMPC and bit-stream based CMPC improve the system's reaction speed and reference tracking ability.

3.4 Summary

Model predictive control in continuous time domain also has a wonderful performance as in discrete time domain. Without the numerical sensitivity and sample rate selection, it becomes even more powerful [8, 10].

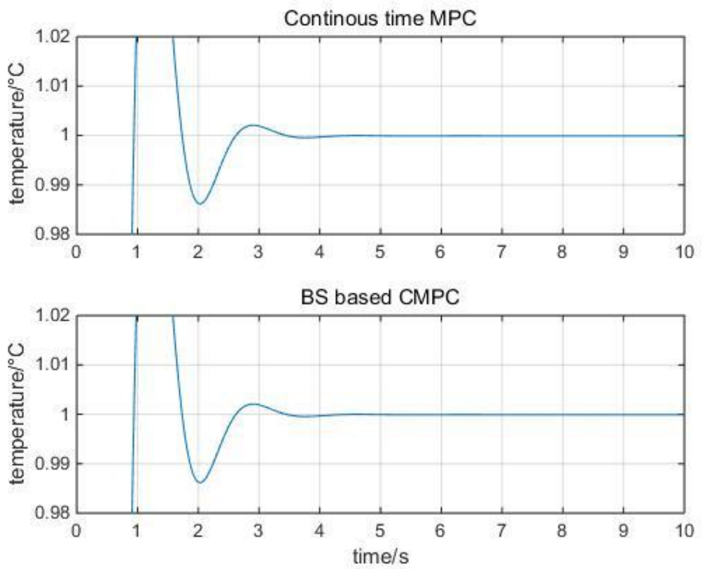

Figure 3.6. Ripples of linear MPC and Bit-stream based linear MPC

Figure 3.6 shows the ripples of bit-stream based linear MPC compared to the ordinary MPC. We could hardly find the difference between these two controllers. This means that in continuous time domain, the performance of bit-stream based MPC becomes better as the switch could work in high frequency in order to reduce the noise.

Chapter 4

Continuous Time Bit-Stream based Model Predictive Control for Nonlinear System

4.1 Introduction

Model predictive control is now one of the most popular method in control engineering. In chapter 3, Continuous time model predictive controller for linear system was introduced and the simulation results shows that linear CMPC has a good performance in control stability and reference tracking.

However, linear systems, in most instances, are simplified models in ideal conditions of practice systems. Most of the systems in practice are nonlinear. In this situation, the linear controllers are inadequate even for moderately nonlinear processes [11]. Several results have been reported in recent past to extend the linear MPC to nonlinear systems [12, 13, 14].

In this chapter, the nonlinear CMPC algorithm is introduced and applied on two classic nonlinear systems. After that, Delta-Sigma modulators are adopted to obtain a bit-stream based nonlinear MPC. Both nonlinear MPC and nonlinear bit-stream based MPC are simulated to investigate the performance of the controllers.

4.2 CMPC for nonlinear system

4.2.1 Control Law

Consider a nonlinear system which is described by the state space model:

$$\begin{cases} \dot{x} = F(x,u) \\ y = G(x) \end{cases} \tag{4.1}$$

and assume that the function F and G have $N_y th$ derivation. Following the similar steps as the linear CMPC controller, we could express the output vector Y as

$$Y(t) = O\big(x(t), U(t)\big) \tag{4.2}$$

where O is output derivative vector and

$$Y(t) = \begin{bmatrix} y & y^{[1]} & y^{[2]} & \cdots & y^{[N_y]} \end{bmatrix}^T$$

The predictor of the nonlinear CMPC is designed as an equation of a variable time T into the future. By expanding time T using Taylor series to $N_y th$ order, we could obtain

$$T(\tau) = \begin{bmatrix} 1 & \tau & \dfrac{\tau^2}{2!} & \cdots & \dfrac{\tau^{N_y}}{N_y!} \end{bmatrix} \tag{4.3}$$

Therefore the output at a future time τ becomes

$$Y(\tau, t) = T(\tau)Y(t) \tag{4.4}$$

After setting the initial time τ_1 and final time τ_2, the time τ is selected into the scale $\lambda \in (0,1)$ as

$$\tau = \tau_1 + \lambda(\tau_2 - \tau_1) \tag{4.5}$$

Similarly as the CMPC, the reference trajectory approximation equation is given as

$$W(t) = R_0 y(t) + R(w(t) - y(t)) \tag{4.6}$$

Also

$$W(t) = R_0 w(t) \tag{4.7}$$

The optimization problem can now be considered as the minimization with respect to U of the cost function J.

$$\begin{aligned} J &= \int_0^1 \left[y(\tau,t) - w(t) \right]^T \left[y(\tau,t) - w(t) \right] d\lambda \\ &= \int_0^1 \left[O\left(x(t),U(t)\right) - W \right]^T T^T(\tau) T(\tau) \left[O\left(x(t),U(t)\right) - W \right] d\lambda \\ &= \left[O\left(x(t),U(t)\right) - W \right]^T T(\tau_1,\tau_2) \left[O\left(x(t),U(t)\right) - W \right] \end{aligned} \tag{4.8}$$

where

$$T(\tau_1,\tau_2) = \int_0^1 T^T(\tau) T(\tau) d\lambda$$

The elements of the matrix $T(\tau_1,\tau_2)$ is calculated by

$$T_{ij}(\tau_1,\tau_2) = \frac{\tau_2^{i+j-1} - \tau_1^{i+j-1}}{(i-1)!(j-1)!(i+j-1)} \tag{4.9}$$

This optimization problem needs to be solved numerically for *u(t)* at each time instant. This usually involves a computational delay and is a major issue in implementation of the controller [15].

4.2.2 Simulation Result

a) Rossler System.

The performance of the nonlinear CMPC was simulated considering Rossler system which is described by the equations [16]:

$$\begin{cases} \dot{x}_1 = x_2 - x_3 \\ \dot{x}_2 = x_1 + ax_2 + u \\ \dot{x}_3 = b + x_1x_3 - cx_3 \end{cases} \tag{4.10}$$

In this case a=0.2, b=0.2, c=5.7.

Because u is contained in $\dot{x}_2$, thus let $y = x_2$.

$$\begin{cases} y = x_2 \\ \dot{y} = \dot{x}_2 = x_1 + ax_2 + u \end{cases} \tag{4.11}$$

Choose $N_y=2$, $\tau_1=0$, $\tau_2=1$.

Then

$$O\big(x(t),U(t)\big) = Y(t) = \begin{pmatrix} x_2 \\ x_1 + ax_2 + u \end{pmatrix} \tag{4.12}$$

$$W(t) = R_0 w(t) = \begin{pmatrix} w(t) \\ 0 \end{pmatrix} \tag{4.13}$$

Therefore we could express the cost function

$$J = \Big[O\big(x(t),U(t)\big) - W\Big]^T T(\tau_1,\tau_2)\Big[O\big(x(t),U(t)\big) - W\Big] \tag{4.14}$$

And calculate the *u(t)* to minimize the cost function *J*.

The result is

$$u = 1.5w(t) - x_1(t) - 1.7x_2(t) \tag{4.15}$$

After that we could build the system model in MATLAB/Simulink and get the results of the simulation of both the original plant system and the system with CMPC. Figure 4.1 shows the system models in Simulink.

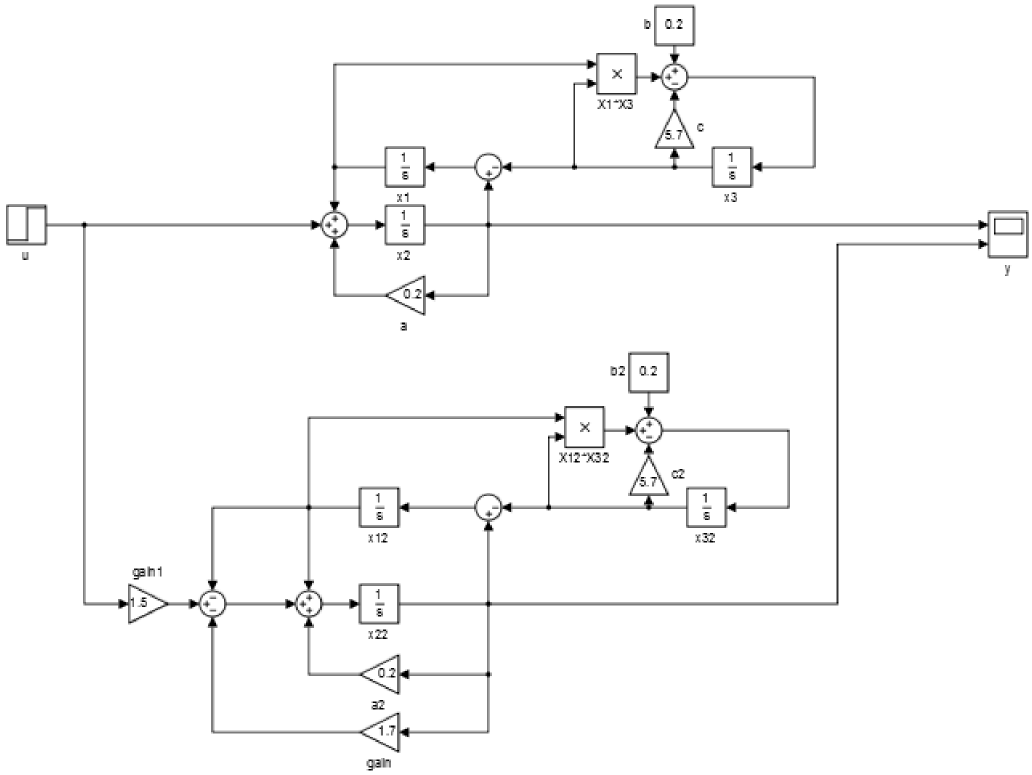

Figure 4.1. System models of Rossler system and CMPC system

Figure 4.2 shows the step response of the two system models.

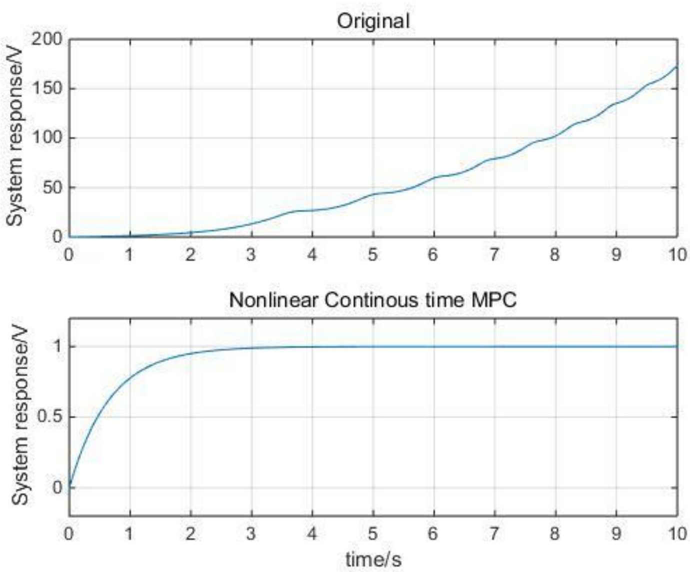

Figure 4.2. Step response of Rossler system and CMPC system

We could see from the figure that the original system step response goes to infinity while the system with CMPC is tracking the reference signal well.

b) Lorenz System

Then consider the Lorenz system which is described by the equations [17, 18]:

$$\begin{cases} \dot{x}_1 = -\sigma x_1 + \sigma x_2 \\ \dot{x}_2 = r x_1 - x_2 - x_1 x_3 \\ \dot{x}_3 = x_1 x_2 - b x_3 + u \end{cases} \tag{4.16}$$

In this case σ=10, r=28, b=8/3.

Because u is contained in $\dot{x}_3$, thus let $y = x_3$.

$$\begin{cases} y = x_3 \\ \dot{y} = \dot{x}_3 = x_1 x_2 - b x_3 + u \end{cases} \tag{4.17}$$

Choose $N_y=2$, $\tau_1=0$, $\tau_2=1$.

Then

$$O\left(x(t), U(t)\right) = Y(t) = \begin{pmatrix} x_3 \\ x_1 x_2 - b x_3 + u \end{pmatrix} \tag{4.18}$$

$$W(t) = R_0 w(t) = \begin{pmatrix} w(t) \\ 0 \end{pmatrix} \tag{4.19}$$

Therefore we could express the cost function

$$J = \left[O\left(x(t), U(t)\right) - W\right]^T T(\tau_1, \tau_2)\left[O\left(x(t), U(t)\right) - W\right] \tag{4.20}$$

And calculate the *u(t)* to minimize the cost function *J*.

The result is

$$u = \frac{1}{2} w + x_3 - x_1 x_2 \tag{4.21}$$

After that we could build the system model in MATLAB/Simulink and get the results of the simulation of both the original plant system and the system with CMPC. Figure 4.3 shows the system models in Simulink.

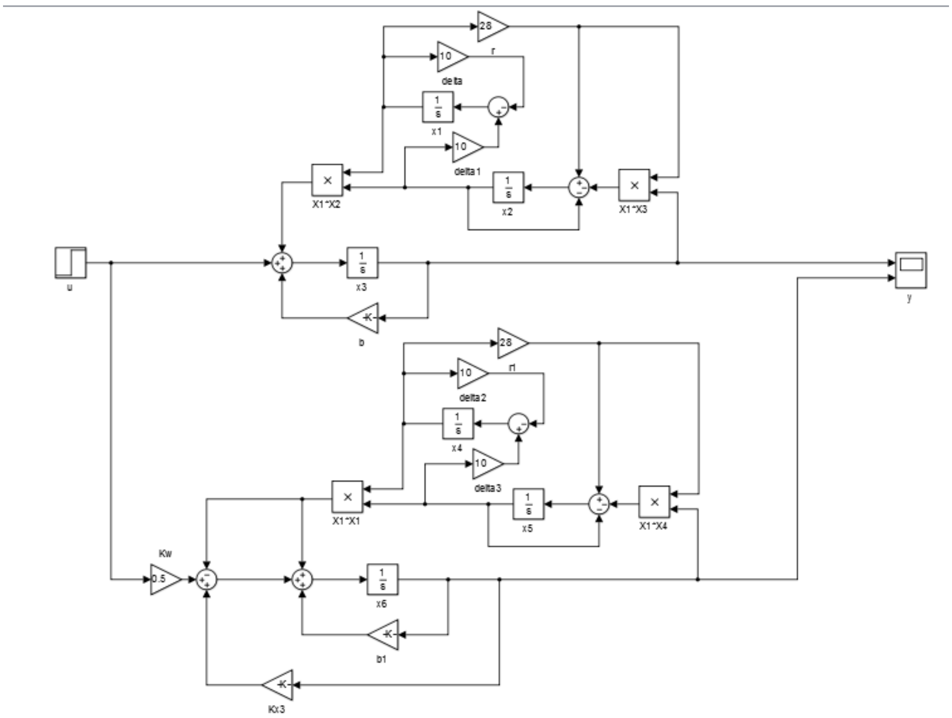

Figure 4.3. System models of Lorenz system and CMPC system

Figure 4.4 shows the step response of the two system models.

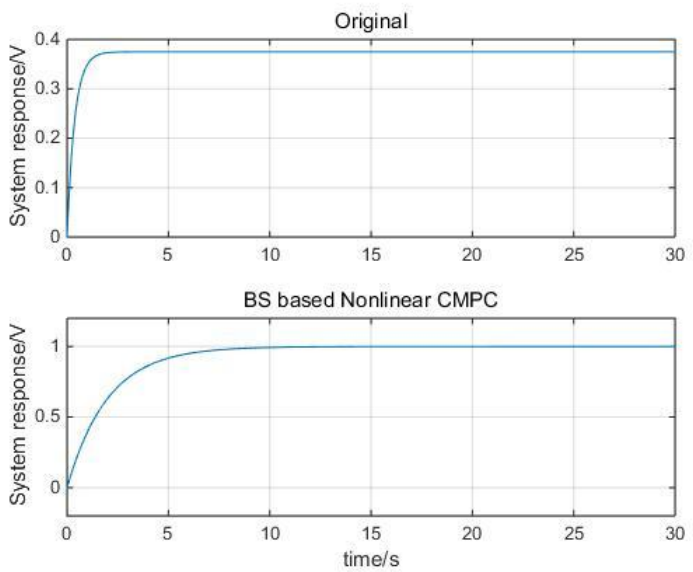

Figure 4.4. Step response of Lorenz system and CMPC system

We could see from the figure that the original system step response cannot follow the input signal while the system with CMPC is following the reference.

4.3 Bit-Stream Based CMPC Control for Nonlinear System

Considering the Rossler system. The CMPC control law is not changed.

In order to convert the control system to a bit-stream based model, Delta-Sigma modulators are adopted between the controller and the system to reduce the communication channel which is shown in figure 4.5.

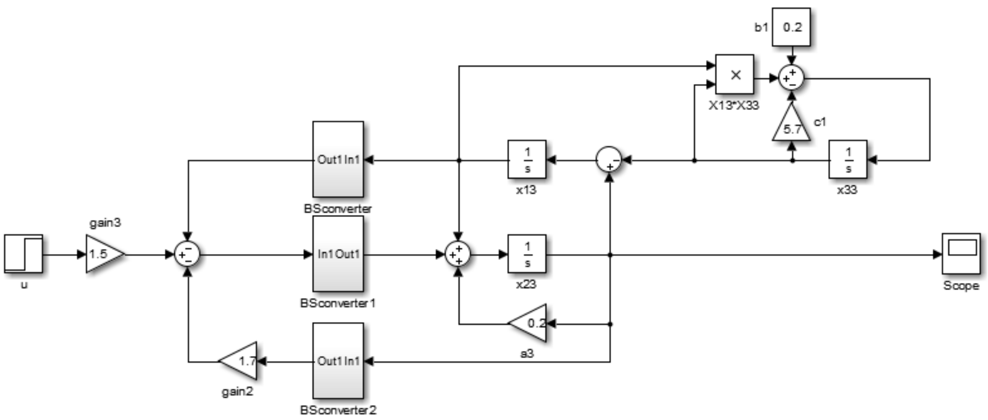

Figure 4.5. Bit-stream based nonlinear CMPC control system for Rossler system

Figure 4.6 shows the step response of the bit-stream based nonlinear CMPC control system.

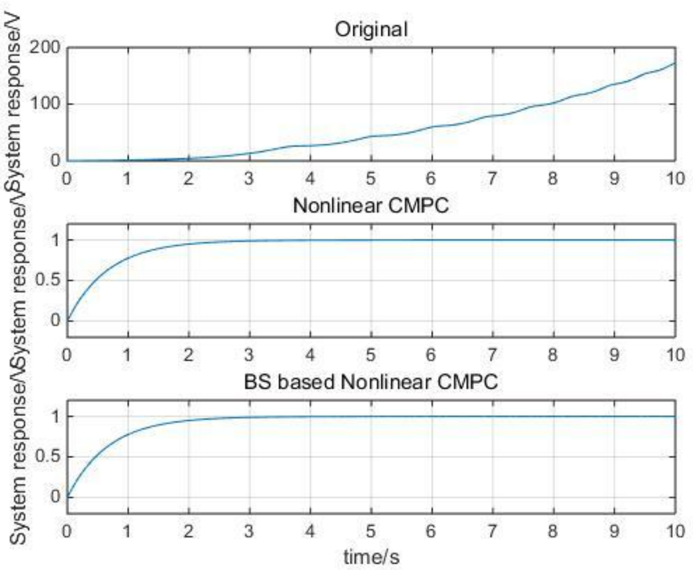

Figure 4.6. Step response of the bit-stream based CMPC control system for Rossler system

Then considering the Lorenz system. The CMPC control law is also not changed.

In order to convert the control system to a bit-stream based model, Delta-Sigma modulators are adopted between the controller and the system to reduce the communication channel which is shown in figure 4.7.

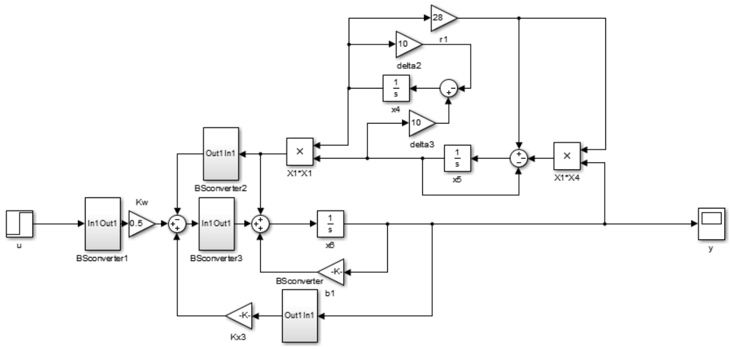

Figure 4.7. Bit-stream based nonlinear CMPC control system for Lorenz system

Figure 4.8 shows the step response of the bit-stream based nonlinear CMPC control system.

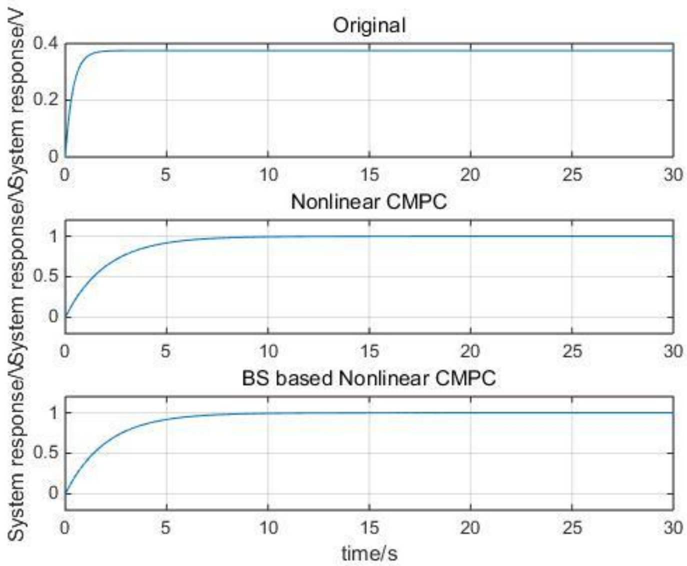

Figure 4.8. Step response of the bit-stream based CMPC control system for Lorenz system

Compare the results between the plant systems, nonlinear CMPC control systems and bit-stream based nonlinear CMPC control systems, we could find that the system keeps its stability and performance after adopting the

Delta-Sigma modulator and both CMPC and bit-stream based CMPC improve the system's reference tracking performance.

4.4 Summary

A state space formulation of nonlinear continuous time MPC has been presented and apply on two classic nonlinear systems, Rossler system and Lorenz system. The simulation results are satisfying. It shows the stability and tracking ability both of MPC and bit-stream based MPC.

The bit-stream based CMPC is also powerful while controlling the nonlinear system as the high frequency of switch would reduce the output noise. In the next chapter, the bit-stream CMPC would be implemented on a DC servo motor via HILINK board to investigate its performance.

Chapter 5

Hardware Implementation

5.1 Introduction

The performance of bit-stream based discrete and continuous time model predictive controllers have been shown in chapter 2, 3 and 4 using simulations based on Matlab toolkit.

Today, with the fast development of computers, most modern advanced control systems are implemented digitally and controlled by computers, for example, the HILINK platform. The HILINK platform offers a seamless interface between physical plants and Matlab/Simulink for implementation of hardware-in-the-loop real-time control systems. It is fully integrated into Matlab/Simulink and has a broad range of inputs and outputs. It allows quick test and iteration of control strategies in real-time with a real plant in the loop. The HILINK platform is a complete and low-cost real-time control system development package and it can be applied for both educational and industrial control systems.

This chapter is organized as follows: the hardware implementation strategy is introduced in the section 5.2 with brief description of the function of the HILINK platform and its realization via Matlab. Section 5.3 gives the model of the DC servo motor and theoretically controls the motor using bit-stream based

MPC. The results of the implementation are included in section 5.4 with summary in section 5.5.

5.2 Hardware Implementation Strategy

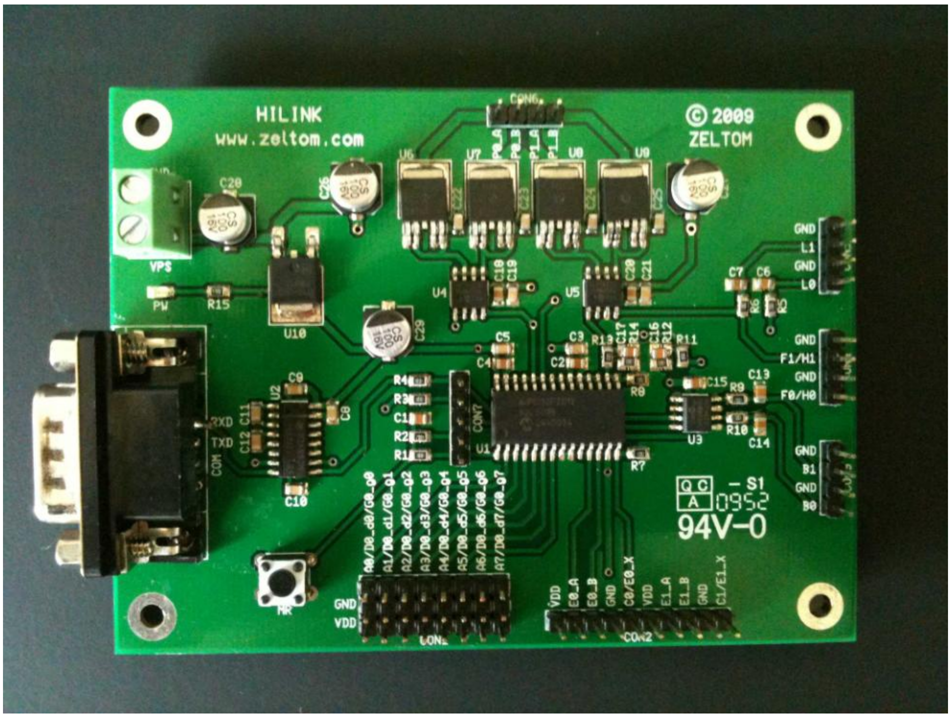

Figure 5.1. The HILINK control board.

The HILINK platform consists of the real-time control board (hardware) and the associated Matlab interface (software). The hardware of the HILINK platform has 8 x 12 bit analog inputs, 2 x 16 bit capture inputs, 2 x 16 bit encoder inputs, 1 x 8 bit digital input, 2 x 12 bit analog outputs, 2 x 16 bit frequency outputs, 2 x 16 bit pulse outputs and 1 x 8 bit digital output. The board also contains two H-bridges with 5 A capability to drive external heavy loads. Some inputs and outputs are multiplexed to simplify the hardware. The board is interfaced to the host computer that runs Matlab through a serial port. The software of the HILINK platform is fully integrated into

Matlab/Simulink/Real-Time Windows Target and comes with Simulink library blocks associated with each hardware input and output. The library contains Analog Input Block, Capture Input Block, Encoder Input Block, Digital Input Block, Analog Output Block, Frequency Output Block, Digital Output Block and Pulse Output Block. The platform achieves real-time operation with sampling rates up to 3.8 kHz.

The hardware implementation uses the HILINK control board to control a DC servo motor via Matlab/Simulink.

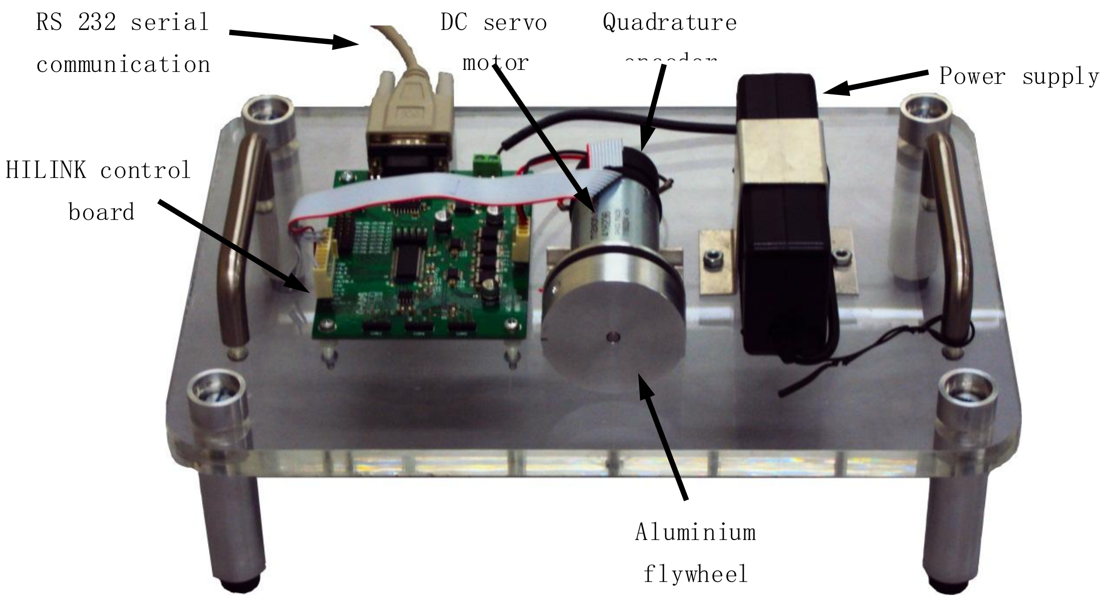

Figure 5.2. DC servo motor platform

The DC servo motor receives up to 12V from the power supply, and is capable of rotating in both directions at a maximum speed of approximately 515 rad/s. A marker point on the flywheel can be used to visually determine the angular position. MATLAB and Simulink will enable the system to be controlled in real-time. Simulink Real-Time Windows Target will be used to compile Simulink blocks into real-time code and communicate with the

HILINK control board. Special input/output blocks are used to communicate between the Simulink blocks and the hardware (green colored blocks in Figure 5.3). A low-pass filter has also been applied to the speed measurement output (block C1). This is because the unfiltered shaft speed is measured at revolutions per sample time, which is too coarse for some speed settings.

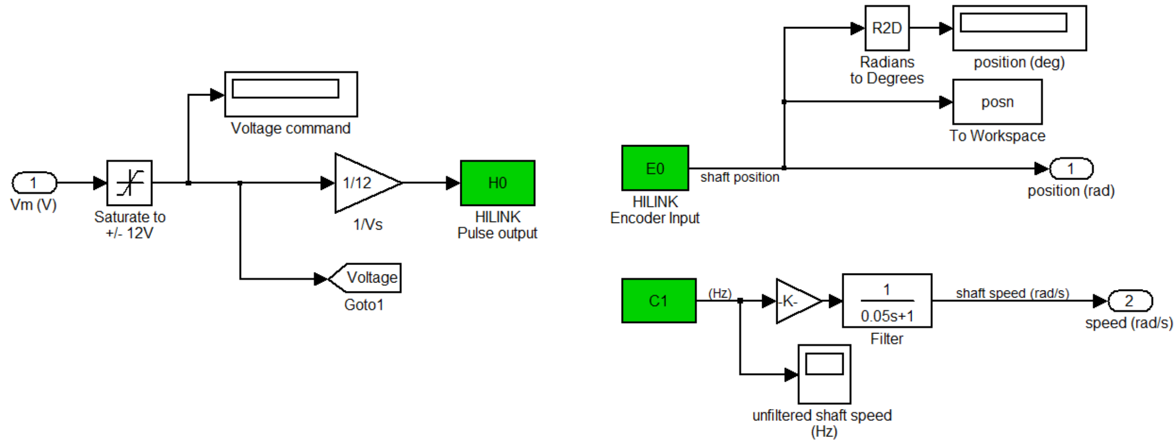

Figure 5.3 HILINK input/output Simulink blocks.

5.3 Theoretical Control Simulation

The speed response model of the DC servo motor is shown below as a first-order linear transfer function.

$$G(s) = \frac{Y(s)}{U(s)} = \frac{41}{0.2s+1} \tag{5.1}$$

In this implementation, a bit-stream based MPC would be designed to control the position of the DC servo motor. Thus, the system model would be extended to:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{41}{0.2s+1} \cdot \frac{1}{s} = \frac{41}{0.2s^2+s} \tag{5.2}$$

which indicates the position response.

Firstly, transform the transfer function to a state space function.

Because:

$$G(s)=\frac{Y(s)}{U(s)}=\frac{Y(s)}{Z(s)}\cdot\frac{Z(s)}{U(s)}=41\cdot\frac{1}{0.2s^2+s} \tag{5.3}$$

Thus

$$\begin{cases} Y(s)=41Z(s) \\ U(s)=(0.2s^2+s)Z(s) \end{cases} \tag{5.4}$$

Make Inverse Laplace Transform,

$$\begin{cases} y=41z \\ u=0.2\ddot{z}+\dot{z} \end{cases} \tag{5.5}$$

Assume

$$\begin{cases} x_1=z \\ \dot{x}_1=x_2=\dot{z} \\ \dot{x}_2=\ddot{z}=5u-5\dot{z}=-5x_2+5u \\ y=41z=41x_1 \end{cases} \tag{5.6}$$

So we could get the system state space function:

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}=\begin{bmatrix} 0 & 1 \\ 0 & -5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}+\begin{bmatrix} 0 \\ 5 \end{bmatrix}u \\ y=\begin{bmatrix} 41 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{cases} \tag{5.7}$$

Therefore, we could calculate the control law of CMPC of the system:

$$\begin{cases} k_w=0.0198 \\ k_x=\begin{bmatrix} 0 & 0.4522 \end{bmatrix} \\ k_y=1 \end{cases} \tag{5.8}$$

$$u=k_w[w-k_y y]+k_x\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{5.9}$$

However, we can’t get the state variable X in hardware implementation.

Note that

$$k_x \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0.4522x_2 = 0.4522\dot{x}_1 = 0.0110 \cdot (41\dot{x}_1) = 0.0110\dot{y} \tag{5.10}$$

And because y indicates the position of the DC motor and thus $\dot{y}$ indicates the speed of the DC motor.

And finally after getting the control law of CMPC, we could get the bit-stream based CMPC of the DC servo motor:

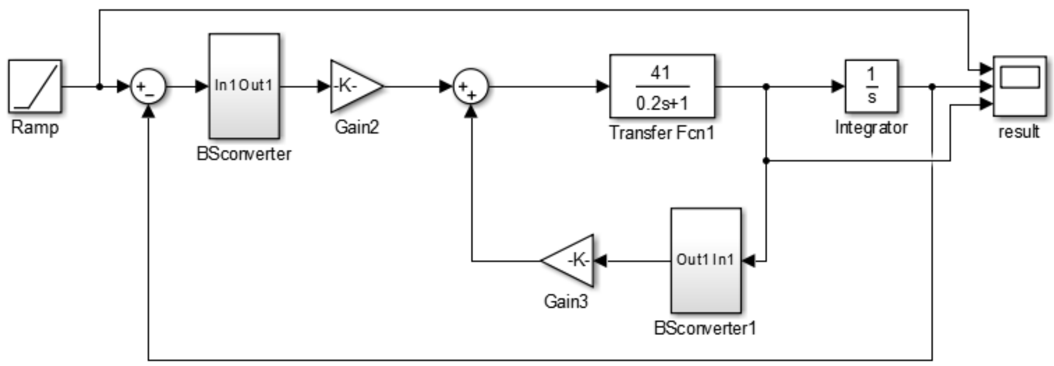

Figure 5.4 Simulation of Bit-stream based CMPC of DC servo motor

Figure 5.5 shows the result of the simulation and the position is following the reference and we can also see the change of the speed.

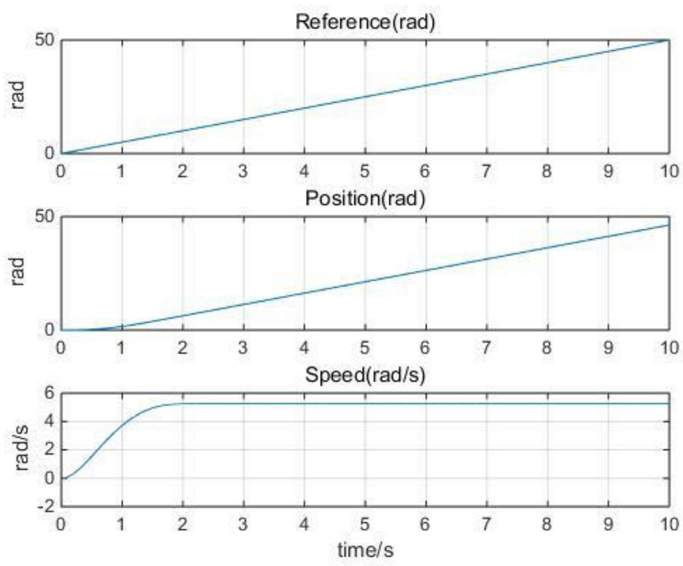

Figure 5.5 Simulation result of DC motor control

5.4 Implementation Results

After successfully controlling the DC servo motor in Simulink, next the bit-stream based CMPC is implemented considering the DC servo motor system.

The close-loop system is shown in figure 5.6 and the block Servo Motor System has already been shown in figure 5.3.

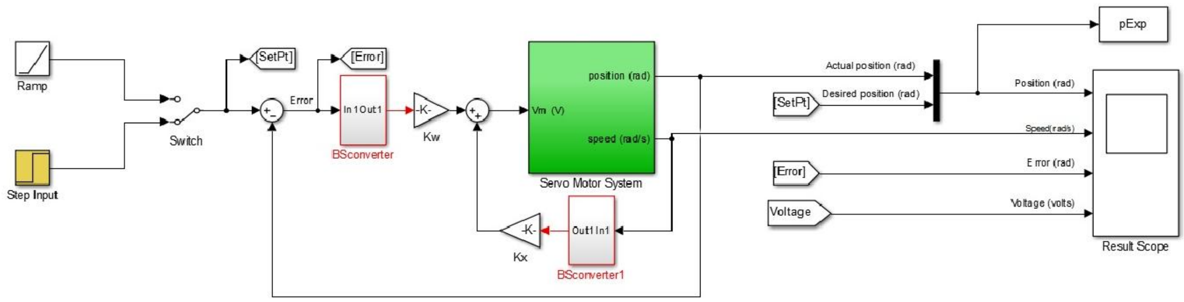

Figure 5.6 the real-time target control for DC motor

For the hardware implementation, several different kinds of references have been used to examine the performance of the bit-stream based CMPC.

Figure 5.7 shows the step response of the system.

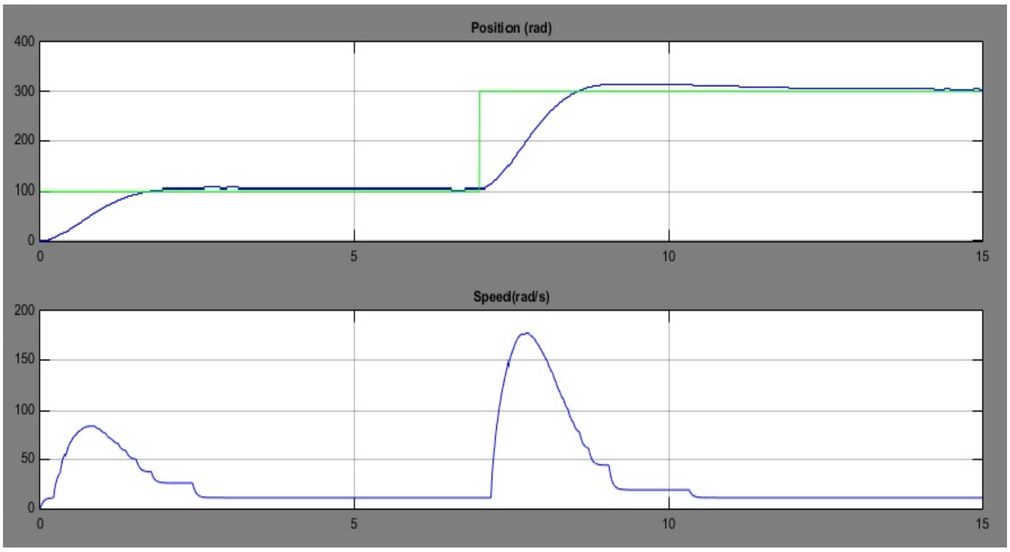

Figure 5.7 Step reference witch X-axis defined by time (S)

The green line indicates the reference and we could see that with the reference changing, the position output follows the reference.

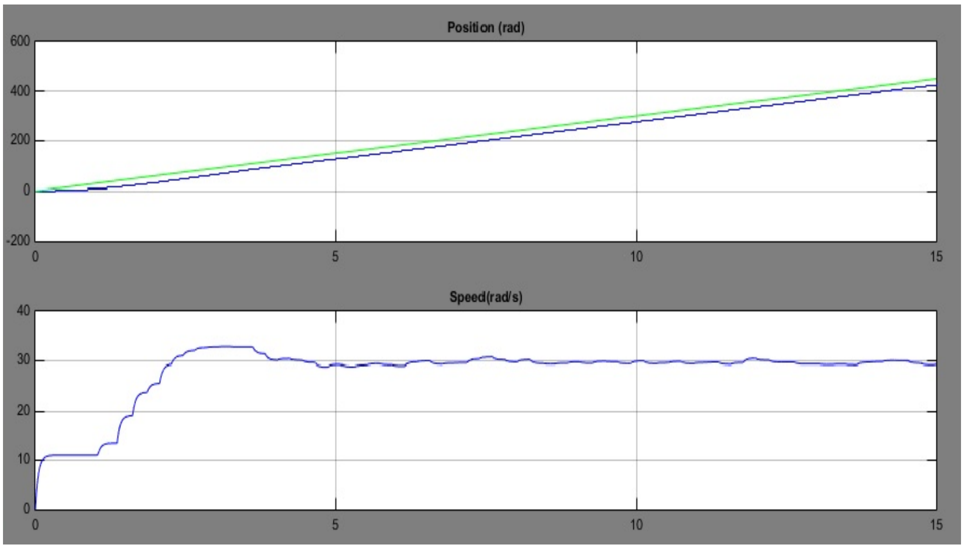

Figure 5.8 Ramp reference witch X-axis defined by time (S)

Figure 5.8 shows the result of the position output while adopting a ramp reference. We could see that the output follows the reference with a small time delay.

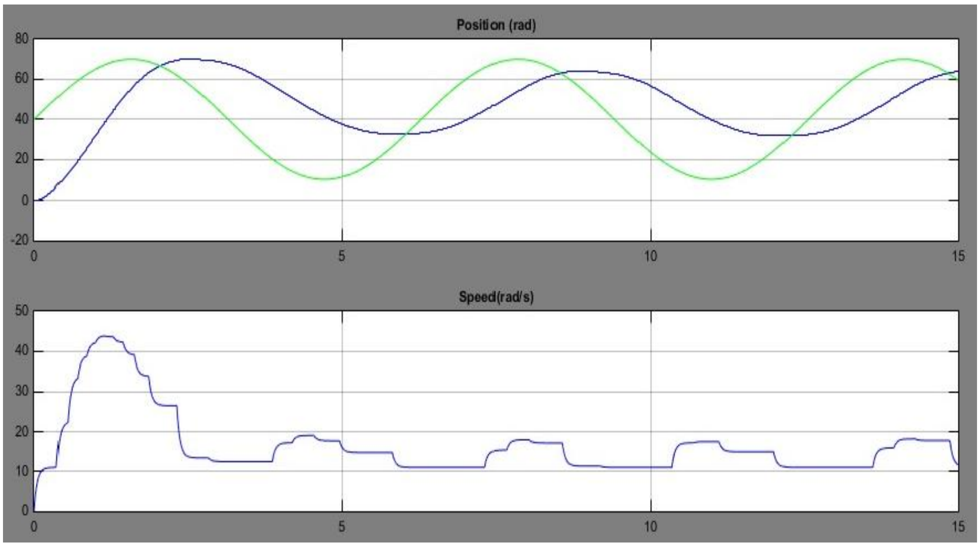

Figure 5.9 Sine wave reference witch X-axis defined by time (S)

Figure 5.9 shows the position output while using a sine wave reference. We could see that there is also a time delay occurring. And the output still follow the reference with a constant time delay.

5.5 Summary

The performance of the linear bit-stream based continuous time model predictive controller was demonstrated by controlling a real physical plant. The physical plant consists of a DC servo motor and a HILINK control board.

The system is to drive the motor under 12 voltage and control its position to follow the given reference. In the hardware implementation, the system cannot ideally have no noise as the simulation in Matlab. Thus a low pass filter is adopted to reduce the ripples of the speed output.

However, because of the switch in the bit-stream modulator, the output would inevitably have some ripples. But this noise has a much smaller influence on the position than on the speed and that's the reason we chose to control the position output of the DC servo motor.

With different kinds of reference, the performance of bit-stream based CMPC is commendable. While tracking a dynamic reference, how to further reduce the delay of the system output worth a deep study.

Chapter 6

Conclusions

Model predictive control has made a significant impact on industrial process control systems since it was originated in the late seventies. It was brought up to solve the existing problems of the traditionally used self-tuning control such as lacking robustness. With over forty years' development, Model predictive control gradually becomes the most important approach to the advanced control of complex industrial processes.

However, traditional MPC is too complex to be implemented in real-time embedded systems. Especially recent years, networked control systems are widely applied in industrial field. In a networked control system there may be very limited resources, in this situation bit-stream technique is put forward. It converts either analog or multi-bits digital signal into bit-stream (single-bit) output through a Delta-Sigma (Δ-Σ) modulator. Bit-stream signal processing is mainly proposed to reduce the silicon consumption and the physical areas for routing bit-parallel signals in digital integrated circuits such as FPGA and VLSI. Furthermore, bit-stream technique reduces the number of interface channels between the subsystems from multiple to single channel, thereby consumes significantly less hardware resources compared to traditional multi-bit processing. In control and power electronics applications, a pulse width modulator (PWM) is not needed anymore since the bit-stream signal is like a fine-grained PWM and hence, it can derive DC-DC converters directly.

The study begins with a review of model predictive control in discrete time domain and the bit-stream technique. For further use of bit-stream technique, several toolboxes are developed in Simulink of Matlab. Then the discrete time MPC is adopted with bit-stream technique. The model predictive controller in discrete time domain has several advantages compared with conventional controllers. Also after adding Delta-Sigma modulator the system performance almost doesn't change. We still could find that the bit-stream based MPC has more ripples thus has more noise than MPC. This phenomenon happens because the Delta-Sigma modulator contains a switch with specific operating frequency and that would lead to some noise. In discrete time system, if the switch frequency is different from the system sampling frequency, the system would become unstable. Therefore, the switch frequency of the modulator should be the same as the system sampling frequency, in this case *100Hz,* and thus lower frequency of switch operation would lead to a higher noise.

To overcome the limitations and difficulties associated with discrete time controllers, continuous time approach to controller design was preferred. Therefore the next part of the research begins with designing controllers using continuous time domain. Initially continuous time model predictive controller (CMPC) was designed for linear systems based on the state space model of the system and combined with bit-stream technique. In continuous time domain, the performance of bit-stream based MPC becomes better as the switch could work in high frequency in order to reduce the noise.

In practice most of the systems are nonlinear to some extent. Therefore the study then focuses on the design of CMPC for nonlinear systems based on state space models of the system. And also bit-stream technique is used on the NLCMPC. The bit-stream based CMPC still has a good performance while controlling the nonlinear system as the switch could work in a high frequency to reduce the output ripple.

The last phase of the research deals with hardware implementation of bit-stream based CMPC using HILINK. The HILINK platform offers a seamless interface between physical plants and Matlab/Simulink for implementation of hardware-in-the-loop real-time control systems. It is fully integrated into Matlab/Simulink and has a broad range of inputs and outputs. It allows quick test and iteration of control strategies in real-time with a real plant in the loop. A DC servo motor has been considered for such implementation. Because of the switch in bit-stream modulator, the output speed is more sensitive with this noise, in this case the position is chosen to be controlled. With different kinds of reference, the performance of bit-stream based CMPC is commendable. While tracking a dynamic reference, how to further reduce the delay of the system output worth a further research.

REFERENCES

[1] C. E. Garcia, D. M. Prett, and M. Morari, “Model predictive control: Theory and practice–a survey,” *Automatica*, vol. 25, pp. 335–348, 1989.

[2] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, “Generalized predictive control–Part I. the basic algorithm,” *Automatica*, vol. 23, pp. 137–148, 1987.

[3] J. Richalet, A. Rault, J. L. Testud, and J. Papon, “Model predictive heuristic control: Applications to industrial processes,” *Automatica,* vol. 14, pp. 413–428, 1978.

[4] C. T. P. S. Clarke, D. W. Mohtadi, “Generalized predictive control–part II extensions and interpretations,” *Automatica,* vol. 23, pp. 149–160, 1987.

[5] Dhafer Al-Makhles, Nitish Patel, Akshya Swain, “Conventional and Hybrid Bit-stream in Real-Time System”, *Intelligent Solutions in Embedded Systems (WISES), 2013 Proceedings of the 11th Workshop on*, 2013

[6] V. Utkin, “Sliding mode control design principles and applications to electric drives,” *Industrial Electronics, IEEE Transactions on*, vol. 40, no. 1, pp. 23 –36, feb 1993.

[7] Ng, Chiu-wa, “Bit-stream signal processing on FPGA”, *The University of Hong Kong (Pokfulam, Hong Kong)*, 2009

[8] H. Demircioglu and P. J. Gawthrop, “Continuous-time generalized predictive control (CGPC),” *Automatica,* vol. 27, pp. 55–74, 1991.

[9] H. Demircioglu and P. J. Gawthrop, “Multivariable continuous-time generalized predictive control (MCGPC),” *Automatica*, vol. 28, pp. 697–713, 1992.

[10] H. Demircioglu and D. W. Clarke, “CGPC with guaranteed stability properties,” *Control Theory and Applications, IEE Proc. D.*, vol. 139, pp. 371–380, 1998.

[11] M. Soroush and C. Kravaris, “Discrete-time nonlinear controller synthesis by input/output linearization,” *AIChE Journal*, vol. 38, pp. 1923–1945, 1992.

[12] P. J. McLellan, T. J. Harris, and D. W. Bacon, “Error trajectory descriptions of nonlinear controller design,” *Chem. Eng. Sci.*, vol. 45, p. 3017, 1990.

[13] C. Kravaris and J. C. Kantor, “Geometric methods for nonlinear process control : 2 control synthesis,” *Ind. Eng. Chem. Res.*, vol. 29, p. 2311, 1990.

[14] C. Kravaris and Y. Arkun, “Geometric nonlinear control - an overview,” *Proc. Of 4th int. conference on chemical process control*, pp. 477–515, 1991.

[15] W. H. Chen, D. J. Ballance, and J. O. Reilly, “Model predictive control of nonlinear systems: Computational burden and stability,” *IEE Proc. Control Theory and Applications*, vol. 147, pp. 387–394, 2000.

[16] O. E. Rossler, “An equation for continuous chaos,” *Physics Lett. A*, vol. 57, pp. 397–398, 1976.

[17] T. L. Vincent and J. Yu, “Control of a chaotic system,” *Dynamics and Control*, vol. 1, pp. 35–52, 1990.

[18] F. Mossayebi, H. K. Qammar, and T. T. Hartley, “Adaptive estimation and synchronization of chaotic systems,” *Physics Lett. A*, vol. 161, pp. 255–262, 1991.

[19] Kannan M. Moudgalya, “Digital Control”, *Chichester, England; Hoboken, NJ: John Wiley & Sons*, c2007.

[20] Liuping Wang, “Model Predictive Control System Design and Implementation Using MATLAB”, *Springer*, 2009.

[21] Dhafer Al-Makhles, Akshya Swain, Nitish Patel, ”Delta-Sigma Based Bit-stream Controller for a D.C.

Motor”, *TENCON 2012 - 2012 IEEE Region 10 Conference*, 2012

[22] Dhafer Al-Makhles, Nitish Patel, Akshya Swain, “A Two-loop Linear Control utilizing ΔΣ modulator”, *Intelligent Solutions in Embedded Systems (WISES), 2013 Proceedings of the 11th Workshop on*, 2013

[23] Dhafer Al-Makhles, Nitish Patel, Akshya Swain, “Bit-stream control system: Stability and experimental application”, *Applied Electronics (AE), 2013 International Conference on*, 2013

[24] Dhafer Al-Makhles, Akshya Swain, Nitish Patel, “Adaptive quantizer for networked control system”, *Control Conference (ECC), 2014 European*, 2014

[25] Richard Schreier, Gabor C. Temes, “Understanding Delta-Sigma Data Converters”, *IEEE transactions on microwave theory and techniques*, 2002

[26] S. Qin and T. Badgwell, “An overview of nonlinear model predictive control applications,” In F. Allgower and A. Zheng: “Nonlinear model predictive control”, *Birkhauser Verlag,* pp. 369–392, 2000.

[27] E. F. Camacho and C. Bordons, Model predictive control in the process industry. London; New York: *Springer-Verlag,* 1995.

[28] P. J. Gawthrop, H. Demircioglu, and I. I. Siller-Alcala, “Multivariable continuoustime generalised predictive control: a state-space approach to linear and nonlinear systems,” *Control Theory and Applications, IEE Proc. -*, vol. 145, pp. 241–250, 1998.

[29] A. W. Ordys and D. W. Clarke, “A state-space description for GPC controllers,” *Int. J. of Systems Science*, vol. 24, pp. 1727 – 1744, 1993.

[30] M. Sun, L. Tian, S. Jiang, and J. Xu, “Feedback control and adaptive control of the energy resource chaotic system,” *Chaos, Solitons & Fractals*, vol. 32, pp. 1725–1734, 2007.

[31] S. Qin and T. Badgwell, “An overview of industrial model predictive control technology,” *AIChE Journal*, vol. 93, pp. 232–256, 1997.

[32] S. Li, K. Y. Lim, and D. G. Fisher, “A state space formulation for Model Predictive Control,” *AIChE Journal*, vol. 35, p. 241, 1989.

[33] H. Fujisaka, M. Segawa, M. Kurosawa, K. Oka, and T.Higuchi, “A control systems with single-bit digital signal processing,” *Proc. of 4th Int. Conf. on Control. Automation. Robotics and Vision*, pp. 1992–1996, Dec 1996.

[34] M. K. Kurosawa, M. Kawakami, K. Tojoj, and T. Katagiri, “Single-bit digital signal processing for current control of brushless dc motor,” *Proceedings of the 2002 IEEE International Symposium on*, vol. 140, no. 3, pp. 589–594, 3 2002.

[35] N. Patel, N. S. Kiong, G. Coghill, and A. Swain, “Online implementation of servo controllers using bit-streams,” *TENCON 2005 IEEE Region 10*, pp. 1–6, 2005.

[36] N. Patel, N. S. Kiong, and G. Coghill, “Neural network implementation using bit streams,” *IEEE Transactions on Neural Networks,* vol. 18, no. 5, pp. 1488–1504, 7 2007.

[37] Y. Murahashi, S. Doki, and S. Okuma, “Realization of 1-bit IIR filter based on delta-sigma modulation under consideration of hardware implementation,” *in Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE,* May 2005, pp. 89–94.

[38] D. Xue, Y. Chen, and D. P. Atherton, Linear Feedback Control Analysis and Design with MATLAB. *Society of Industrial and Applied Mathmatics,* 2009.

[39] Y. Murahashi, S. Doki, and S. Okuma, “Hardware realization of novel pulsed neural networks based on delta-sigma modulation with gha learning rule,” *in Circuits and Systems, 2002. APCCAS ‘02. 2002 Asia-Pacific Conference on*, vol. 2, 2002, pp. 157 – 162 vol.2.

[40] Y. Ito, S. Doki, and S. Okuma, “Digital filter design with state space method for 1-bit signal processing based on delta-sigma modulation,” *Electrical Engineering in Japan,* vol. 175, no. 4, pp. 48–56, 12

2011.

[41] D. A. Johns and D. M. Lewis, “Design and analysis of delta-sigma based IIR filters,” *IEEE Transactions on Circuits and Systems,* vol. 40, no. 4, pp. 233–240, April 1993.

[42] H. Fujisaka, R. Kurata, M. Sakamoto, and M. Morisue, “Bit-stream signal processing and its application to communication systems,” *IEE Proc. Circuit Devices,* vol. 149, no. 3, pp. 159–166, June 2002.

[43] Y. F. Chan, M. Moallem, and W. Wang, “Design and implementation of modular FPGA-based PID controllers,” *Industrial Electronics, IEEE Transactions on*, vol. 54, no. 4, pp. 1898 –1906, aug. 2007.

[44] Z. Fang, J. Carletta, and R. Veillette, “A methodology for FPGA-based control implementation,” *Control Systems Technology, IEEE Transactions on,* vol. 13, no. 6, pp. 977 – 987, nov. 2005.

[45] J. Doyle, B. Francis, and A. Tannenbaum, *Feedback Control Theory*. New York: Macmillan, 1992.

[46] H. Sira-Ramirez and R. Silva-Ortigoza, *Control Design Techinques in Power Electronics Devices*. Springer, 2006.

[47] X. Wu, V. A. Chouliaras, J. L. Nunez-Yanez, and R. M. Goodall, “A novel ΔΣ control system processor and its vlsi,” *IEE Proc.-Control Theory Appl.*, vol. 16, no. 3, pp. 217–228, 3 2008.

[48] X. Wu and R. Goodall, “One-bit processing for real-time control,” *IEE Proc.-Control Theory Appl.,* vol. 152, no. 4, pp. 403–410, 7 2005.

[49] J. C. and Candy, “A use of double integration in sigma delta modulation,” *IEEE Trans. Comm.,* vol. 33, pp. 249–258, March 1985.

[50] B. E. Boser and B. A. Wooley, “The design of sigma-delta modulation analog-to digital convertotrs,” *IEEE Journa of Solide-State Circuit,* vol. 23, no. 6, pp. 1298–1308, December 1988.

[51] R. H. Middleton and G. C. Goodwin, *Digital Control and Estimation: A Unified Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[52] R. Goodall and B. Donoghue, “Very high sample rate digital filters using the delta; operator,” *Circuits, Devices and Systems, IEE Proceedings G,* vol. 140, no. 3, pp. 199 –206, jun 1993.

[53] W. Hu, G. Liu, and D. Rees, “Networked predictive control system with data compression,” *in Networking, Sensing and Control, 2007 IEEE International Conference on,* 2007, pp. 52–57.

[54] C. C. de Wit, F. Gomez-Estern, and F. Rubio, “Delta-modulation coding redesign for feedback-controlled systems,” *IEEE Transactions on Industrial Electronics,* vol. 56, no. 7, pp. 2684–2696, 2009.

[55] T. Li and Y. Fujimoto, “Control system with high-speed and real-time communication links,” *IEEE Transactions on Industrial Electronics,* vol. 55, no. 4, pp. 1548–1557, 2008.

[56] R. Brockett and D. Liberzon, “Quantized feedback stabilization of linear systems,” *IEEE Transactions on Automatic Control,* vol. 45, no. 7, pp. 1279–1289, 2000.

[57] E. Monmasson, L. Idkhajine, and M. Naouar, “FPGA-based controllers,” *IEEE Industrial Electronics Magazine,* vol. 5, no. 1, pp. 14–26, march 2011.

[58] E. Monmasson and M. Cirstea, “FPGA design methodology for industrial control systems,” *IEEE Transactions on Industrial Electronics*, vol. 54, no. 4, pp. 1824 –1842, Aug 2007.

[59] S. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters:Theory, Design, and Simulation*. Wiley-IEEE Press, 1997.

[60] H. Demircioglu and P. J. Gawthrop, “Multivariable continuous-time generalized predictive control (MCGPC),” *Automatica*, vol. 28, pp. 697–713, 1992.

[61] S. Li, K. Y. Lim, and D. G. Fisher, “A state space formulation for Model Predictive Control,” *AIChE Journal*, vol. 35, p. 241, 1989.

[62] K. S. Park, J. M. Joo, J. B. Park, C. Ho, and T. S. Yoon, “Control of discrete-time chaotic systems using generalized predictive control,” 1997.

[63] I. Serdar, “Support vector machines based generalized predictive control of chaotic systems,” *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng)*, vol. E89-A, pp. 2787–2794, 2006.

[64] A. Boukabou and N. Mansouri, “Prediction-based control of continuous-time chaotic systems,” *ARISER*, vol. 3, pp. 99–106, 2007.

[65] O. E. Rossler, “An equation for hyperchaos,” *Physics Letters A*, vol. 71, pp. 155–157, 1979.

[66] M. Soroush, “Nonlinear state-observer design with application to reactors,” *Chemical Engineering Science*, vol. 52, pp. 387–404, 1997.

[67] C. T. Chen, *Linear System Theory and Design*. Oxford University Press, Inc.,1998.

[68] M. Zeitz, “The extended luenberger observer for nonlinear systems,” Syst. Control Lett., vol. 9, pp. 149–156, 1987.

[69] P. H. Wallman, “Reconstruction of unmeasured quantities for nonlinear dynamic processes,” Ind. Engng Chem. Fundam, vol. 18, p. 327, 1979.

[70] R. S. H. Istepanian and J. F. Whidborne, Digital Controller Implementation and Fragility: A Modern Perspective. Springer, 2001.

[71] K. Z. Mao and T. Y. Chai, “Model predictive control algorithms for systems with slow sampled outputs,” IEE Proc. Control Theory and Applications,, vol. 143, pp. 551–556, 1996.

[72] Feedback Instruments Limited. (1996), Process Trainer PT326. User Manual: Crowborough.

[73] J. Eyre and J. Bier, “The evolution of DSP processors,” IEEE Signal Processing Mag., vol. 17, pp. 43–51, 2000.

[74] dSPACE Inc. (2005), DS1104 R&D Controller Board Reference Manual. User Manual.

[75] K. Ogata, System Dynamics. Prentice Hall, 1992. REFERENCES 98

[76] R. Pintelon and J. Schoukens., System Identification: A Frequency Domain Approach. USA: IEEE Press, 2001.

[77] R. D. Nowak and B. D. Van Veen, “Random and pseudorandom inputs for volterra filter identification,” IEEE Trans. Signal Process., vol. 42, pp. 2124–2135, 1994.

[78] K. Dutton, S. Thompson, and B. Barraclough, The Art of Control Engineering. Great Britain: Addison-Wesley, 1997.

[79] K. Ogata, Discrete-Time Control Systems. Prentice Hall, 1995.