Master's Thesis 2013

Candidate: Ramanathan Annamalai

Title: Discrete time Linear Quadratic (LQ) optimal control Vs MPC: Integral action and handling constraints

Telemark University College
Faculty of Technology
Kjølnes
3914 Porsgrunn
Norway

Telemark University College
Faculty of Technology
M.Sc. Programme

MASTER'S THESIS, COURSE CODE FMH606

Student:	Ramanathan Annamalai, 113824
Project title:	Discrete time Linear Quadratic (LQ) optimal control Vs MPC: Integral action and handling constraints
Signatures:	……
Number of pages:	120
Keywords:	**LQ Optimal control, MPC, Quadruple tank process, Constraints, Kalman filter and PI control**
Supervisor:	**David Luigi Di Ruscio** Sign.: ……………………….
Censor:	Sign.: ……………………….
Availability:	Open

Archive approval (supervisor signature): sign.: …………………………………… **Date :** ………………………

Abstract

Studies on optimal control strategy have been discussed for long years by academic institutions and by industrial persons. This thesis contributes to this wide range of study and compares the Linear quadratic optimal control and Model Predictive control based on constraints handling. MPC is much more popular and used controller than LQ optimal controller and comparison between these controllers are done based on their performance to reach the set point and constraints handling.

Theoretical study and literature overview of LQ and MPC is provided and also theoretical description on how constraints are handled. A non linear process like quadruple tank system is selected to compare the performance of these controllers. Quadruple tank system is a multiple input multiple output, contains unknown slowly varying process and measurement disturbance. Minimum phase and Non-minimum phase of the quadruple tank also discussed based on placement of zero. LQ optimal controller is implemented in the quadruple tank system, in two forms such that constrained using if else loops and unconstrained. MPC controller is implemented in three forms such that algorithm based constraints, if else loop constraints and unconstrained form.

Comparisons are performed within LQ control, within MPC controller and also between constraints handling of LQ and MPC. PI control was also implemented using RGA analysis for comparison. Kalman filter was used to predict the state of unmeasured tank level.

It is observed that MPC unconstrained reaches the set point much quicker, but it violates the constraint limits. MPC algorithm based constraint handling reaches the set point much faster than other controller, it is stable, and robust. MPC if else constraint also reaches the set point at the same time, but it has some overshoot. LQ optimal controller reaches the set point later than MPC but earlier than PI. Finally PI takes a long time to reach the set point.

Table of Contents

1. Introduction 10

2. Linear Quadratic Optimal Control 13

2.1 Discrete Linear Quadratic Optimal Control 13

2.2.1 Pontryagin Maximum Principle 14

2.1.2 Discrete optimal control of linear systems 15

2.1.3 Discrete time Algebraic Riccati equation 17

2.2 Integral Action in Discrete LQ Optimal Controller 18

3. Model Predictive Control 21

3.1 Introduction 22

3.1.1 Definitions 22

3.1.2 Theory 24

3.2 Model Predictive control with Integral Action 28

4. Handling Constraints 30

4.1 Classification of Constraints 30

4.2 Types of Constraints 31

4.3 Constraints Handling in MPC 32

4.4 Constraints Handling in LQ Optimal Control 33

5. Kalman Filter 34

6. PID 35

6.1 RGA Analysis 35

7. Problem Formulation 38

7.1 Four Tank Level Process 38

7.2 Physical Model 40

7.3 Minimum Phase 43

7.4 Non-Minimum Phase 44

7.5 Properties of Linearized Model 44

8. Problem Solution (Matlab Simulation) 46

8.1 LQ Optimal Control (Constraints handling Comparison) 47

8.1.1 Minimum Phase system Comparison 50

8.1.2 Non-Minimum Phase system Comparison 53

8.2 MPC (Constraints Handling Comparison) 56

8.2.1 Minimum Phase 59

8.2.2 Non-Minimum Phase 63

8.3 LQ, MPC and PI comparison 67

9 Future Developments 73

10 Conclusion 74

Reference 75

Appendix 1: Master Thesis task description SIV-53-13 77

Appendix 2: Properties of Linearized Model 79

Appendix 3: LQ with Unconstrained 81

Appendix 4: LQ Constrained (if else loop) 86

Appendix 5: Writing & Reading data from Excel File (Minimum Phase) 92

Appendix 6: Writing & Reading data from Excel File (Non-Minimum Phase) 94

Appendix 7: MPC constrained (Algorithm based) 96

Appendix 8: MPC constrained (if else loop) 102

Appendix 9: MPC unconstrained 108

Appendix 10: Writing and Reading from Excel file for minimum phase, MPC 113

Appendix 11: Writing and Reading from Excel file for non minimum phase, MPC 115

Appendix 12: PI Controller 117

Appendix 13: RGA Analysis for PI controller 120

Appendix 14: CD

Preface

First of all I would like to thank GOD for giving this beautiful life and planet Earth. This thesis work is a mandatory part of master degree in System and Control Engineering at Telemark University College (TUC), Porsgrunn, Norway. This report is written based on the requirement of the tasks given by professor. This report provides a theoretical overview of linear quadratic optimal control and Model predictive control with integral action. Constraints handling are also discussed with respect to various controllers. Comparison in performance of LQ, MPC and PI controllers are made based on Constraint handling on a non linear quadruple tank process which is MIMO and contains unknown slowly varying process. MATLAB was used for simulation.

I especially want to thanks my supervisor Associate Professor David Di Ruscio for his supervision, suggestions, lecture notes, journal paper and all the advice/valuable comments he has given over the whole thesis work.

Special thanks to all those professors and their valuable notes which imparted knowledge in me and made me work on this thesis confidently. I want to extend my thanks to the Library of Telemark University for providing valuable books and access to various journal papers.

I would also like to thank my friends who supported, helped and gave confidence to me. Special thanks to my parents who had confidence in me and encouraged me to take up this Master Degree.

Porsgrunn, 01-06-2013
Ramanathan Annamalai

Nomenclature

This section gives a list of symbols and abbreviations used in the report.

Abbreviation	Meaning/Explanation
LabView	Laboratory Virtual Instrument Engineering Workbench
LQ	Linear Quadratic
LQOC	Linear Quadratic Optimal Control
MATLAB	Matrix Laboratory
MIMO	Multiple Input Multiple Output
MPC	Model Predictive Control
PI	Proportional Integral
PID	Proportional Integral Derivative
RGA	Relative Gain Array

Symbols	Meaning/Explanation
$u_k, u_{k/L}$	Control signal and Control vector
$\Delta u_k, \Delta u_{k/L}$	Change in control signal, change in control vector
$r_k, r_{k+1/L}$	Reference signal and future reference signal
H	Hamiltonian Matrix
K	Kalman Gain
R	Solution to Riccati equation
J_k	Cost function
I	Identity matrix
A, B, C and D	State space matrices
$A_c, B_c, C_c \ and \ D_c$	Continuous time matrices
G	Feedback for optimal controller
P, Q	weighting matrices
K_p, T_i	Proportional Gain and integral time
k_1, k_2	Pump Constant
γ_1, γ_2	Valve constant
T_1, T_2, T_3, T_4	Time constant
n, r, m	Size of A matrices, Size of control matrix and size of D matrix

Overview of figures

Figure 1-1: *Components of Control System (Kwon. and Han, 2005)* 11

Figure 3-1: *Sketch of an MPC Controller(Halvorsen., 2011)* 25

Figure 3-2: *The moving horizon strategy of MPC (Holkar. and L.M.Waghmare, 2010)* 26

Figure 3-3: *Moving horizon representation for MPC (Halvorsen., 2011).* 26

Figure 3-4: *Model Predictive Control Scheme (Nikolaou, 2011).* 27

Figure 6-1: *RGA analyses for minimum and non-minimum phase of quadruple tank system.* 37

Figure 7-1: *Diagram of quadruple-tank process(Johansson, May 2000).* 40

Figure 7-2: *Linear model analysis of quadruple tank system, left shows the minimum phase system and right show non-minimum phase properties.* 45

Figure 8-1: *Various type of simulation performed in MATLAB for quadruple tank process using minimum and non-minimum phase system.* 46

Figure 8-2: *Different types of LQ optimal control simulation implemented on four tank process.* 47

Figure 8-3: *Flow chart of LQ optimal control programming Part I.* 48

Figure 8-4: *Flow chart of LQ optimal control programming part II.* 49

Figure 8-5: *Level of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.* 51

Figure 8-6: *Control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.* 51

Figure 8-7: *Change in control signal ΔU of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller* 52

Figure 8-8: *Level of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller* 54

Figure 8-9: *Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.* 54

Figure 8-10: *Change in control signal ΔU of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.* 55

Figure 8-11: *Different types of LQ optimal control simulation implemented on four tank process.* 56

Figure 8-12: *Flow chart of MPC with integral action programming part I.* 57
Figure 8-13: *Flow chart of MPC with integral action programming part II* 58
Figure 8-14: *Level of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action* 60
Figure 8-15: *Control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action* 61
Figure 8-16: *Change in control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action* 61
Figure 8-17: *Level of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action.* 64
Figure 8-18: *Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action* 65
Figure 8-19: *Change in Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action* 65
Figure 8-20: *Comparison of LQ, MPC and PI control with reference signal for Tank 1. Constrained implemented in all controller. It's a minimum phase quadruple tank process.* ..68
Figure 8-21: *Comparison of LQ, MPC and PI control with reference signal for Tank 2. Constrained implemented in all controller. It's a minimum phase quadruple tank process.* ..69
Figure 8-22: *Comparison of LQ, MPC and PI control, based on constraint limit (0≤U≤5) on control signal U for Tank 1. It's a minimum phase quadruple tank process.* 70
Figure 8-23: *Comparison of LQ, MPC and PI control, based on constraint limit (0≤U≤5) on control signal U for Tank 2. It's a minimum phase quadruple tank process* 70
Figure 8-24: *Comparison of LQ, MPC and PI control, based on constraint limit (-0.4≤ΔU≤0.4) on control signal ΔU for tank 1 and tank 2. It's a minimum phase quadruple tank process.* ..71

Overview of tables

Table 6-1: *Pairing combination for minimum phase and non-minimum phase quadruple tank system* 37

Table 7-1: *Information on Inputs, outputs and states of Quadruple tank process* 39

Table 7-2: *Flows to the tanks generated by the two pumps* 39

Table 7-3: *Common Parameter for Quadruple Tank process(Johansson, May 2000)*............42

Table 7-4: *Operating condition of quadruple tank process* 42

Table 8-1: *Critical parameter for simulation for LQ optimal control for minimum phase system* 50

Table 8-2: *Critical parameter for simulation for LQ optimal control for non minimum phase*53

Table 8-3: *Critical parameter for MPC simulation on minimum phase quadruple tank process.* 59

Table 8-4: *Critical parameter for MPC simulation on non minimum phase quadruple tank process.* 63

Table 8-5: *Constraint limits on all Controller (for PI only U constraints used)* 67

Table 8-6: *Comparison overview of various controllers based on constraint for minimum phase quadruple tank system.* 72

1. Introduction

Controlling a physical process or an industrial process has always been a challenge. Control theory engineering deals with various physical process / dynamic systems, its inputs and set point. Mostly a controller is used to control the physical process. The main part of this thesis is regarding this controller.

The most common controller used is PID Controller. Various types of advance controllers are also used, based on a survey in Japanese industries like steel, power, petrochemical and paper industries the following advance controllers are used (Takatsu et al., 1998)

1) Advance PID
2) Decoupling control
3) Dead time compensation
4) Gain scheduled (G-Schedule)
5) PID Auto Tune
6) LQ Optimal
7) Observer
8) Kalman Filter
9) MPC
10) Adaptive controller
11) $H_{\bowtie}$ Optimal control
12) Rule base control
13) Fuzzy control
14) Application of Neural Networks
15) Repetitive control
16) Exact Linearization for Nonlinear system Control (Exact-Linearized)
17) Sliding mode control
18) Optimizing control

This thesis is focused on advance controllers and more specifically on optimal control theory. Optimization has always played a crucial role in decision processes concerning physical or organization systems. Every time there is a need to make a selection between a set of possible choices, one would like to pick the decision that costs the least, provide an optimal solution, satisfies all constraints and is practical to implement. Therefore, optimization has become an integral part of any scientific and engineering discipline. Optimal control theory is a mature mathematical discipline with numerous applications in both science and engineering (Todorov, 2006). The objective of optimal control theory is to determine the control signals that will cause a process to satisfy the physical constraints and at the same time minimize (or maximize) some performance criterion(Ruscio, 2012c).

Three important parameters of the optimal controls are Model of the system, performance criterion and control structure which can be summarized visually in figure 1.1.

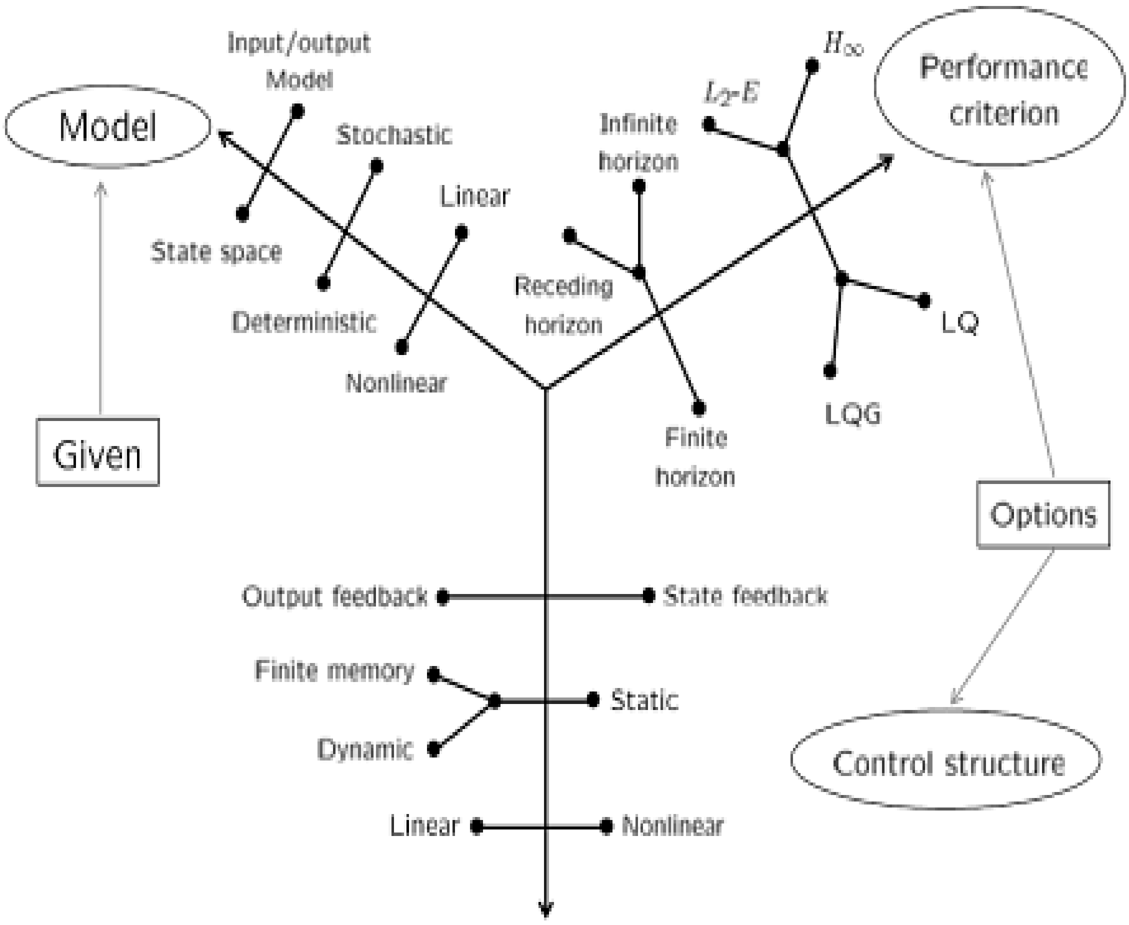


Figure 1-1: *Components of Control System (Kwon. and Han, 2005)*

The core part of the thesis report consists of LQ and MPC as performance criterion, control structure as nonlinear and model as state space. The main contributions of this thesis are itemized below:

- Quadruple tank process used as a benchmark process, which is a Non-Linear process
- LQ optimal control with integral action with constraint (if else loop) and unconstrained forms are implemented on the tank process
- MPC with integral action with constraint and unconstrained forms are implemented on tank process. Also, PI control in velocity form implemented on the non linear process.
- Compare the performance of LQ optimal control based on constraints and unconstrained on a quadruple tank process.

- Compare the performance of MPC control based on constraints and unconstrained on a quadruple tank process.
- Compare the performance of LQ optimal control, MPC and PI Control based on constraints handling on a quadruple tank process.

The thesis is formulated in such way that the flow of information is easy to understand and clear. The further part of thesis is divided into 9 chapters. Chapter 2 theory and formulation of LQ optimal control defined. Chapter 3 theory and formulation of MPC control described. Chapter 4 describes the types of constraint and how constraints are handled on both these controllers. Chapter 5 provides a brief concept of Kalman filter. Chapter 6 describes the PID in velocity form briefly. Problem formulation of the quadruple tank is described on chapter 7. Chapter 8 is the important part of thesis whether the results of simulation are plotted and described in detail. Comparisons are based within LQ optimal control based on constraint and unconstrained and similarly next section describes comparison on MPC controller based on constraint and unconstrained, last section compares the LQ, MPC and PI controller based on constraints. Future developments are discussed on chapter 9 and conclusions are pen downed on chapter 10. Further References are provided and at last Appendix is provided where MATLAB codes for various simulations are shown and a compact disc is attached with this thesis report.

2. Linear Quadratic Optimal Control

This chapter provides an overview of linear quadratic optimal control (LQOC). The main concept behind this method is to provide optimal control i.e. the output should follow the setpoint in the best way. Control design objectives are formulated in terms of cost function. The cost function is quadratic and the main objective is to reduce the cost function and to have infinite prediction horizon (Ruscio, 2012c).

Linear quadratic (LQ) optimal control was pioneered by Kalman, which has been playing a central role in modern control theory. Up to now, deterministic LQ problem has been investigated extensively by many researchers. Stochastic LQ control for the system governed by Ito equation was initiated by Wonham (Huang et al., 2006). The solution of the unconstrained linear quadratic (LQ) optimal control problem is well known (Anderson and Moore, 1989). Several methodologies for solving the linearly constrained case have also been developed (Faybusovich, 1982; Faybusovich and Moore, 1995; Faybusovich and Moore, 1996). Faybusovich and Moore (1995) and Faybusovich and Moore (1996), the interior point methodology (IPM) for solving the quadratic programming problems is extended to the infinite-dimensional setting with complexity estimates similar to the finite dimensional case. When applied to the constrained LQ optimal control problem, in the case of linear or quadratic constraints, the authors show that the optimal control can be obtained by solving a sequence of unconstrained LQ problems together with a sequence of finite dimensional linear algebraic equations.

This chapter is further divided various section and sub section. Discrete linear quadratic optimal control is described in section 2.1, which is further divided into subsections. 2.1.1 Provides valuable information about pontryagin maximum principle, 2.1.2 talks in detail on discrete optimal control of a linear system (non linear system is converted to a linear system) and last subsection 2.1.3 describes the Riccati equations. Section 2.3 provides the valuable information and equation of integral action in linear quadratic optimal control.

2.1 Discrete Linear Quadratic Optimal Control

LQOC can be divided into continuous time linear quadratic optimal control and discrete time linear quadratic optimal control. This thesis will focus more on discrete time linear quadratic optimal control as discrete is more computer friendly and can be easily implemented in software's for computational.

The approach followed here relies on the application of Pontryagin maximum principle to the dynamic linear model, and a quadratic cost function is considered. In continues time, the system to be controlled is assumed to be described by a linear controllable state space model with the state variable for direct measurement. The resulting controller problem consists of a state feedback whose gain *G* depends on the solution *R*, where R is the positive solution to the discrete time algebraic Riccati equation.

2.2.1 Pontryagin Maximum Principle

The classical result of optimal control theory is the Pontryagin Maximum Principle (PMP), which provides necessary optimality conditions for control system governed by differential equations under various constraints (*Mordukhovich and Shvarsman, 2012*). The Discrete Maximum time maximum principle which is a method for solving the discrete time optimal control problem (Ruscio, 2012c). Pontryagin's maximum principle (PMP) states a necessary condition that must hold on an optimal trajectory. It is a calculation for a fixed initial value of the state, x(0) (*University*).

The following ideas and equations are based on Lecture notes "Optimal model based Control" (Ruscio, 2012c).

Consider a discrete time dynamic process model

$$x_{k+1} - x_k = f(x_k \ , u_k \, , k) \tag{1}$$

Equation (1) denotes f(.) as a general non-linear vector function and k as a discrete time. Further we consider an optimal performance index or cost function in a discrete form

$$J_i = S(x_N) + \sum_{k=i}^{N-1} L(x_k \, , u_k) \tag{2}$$

S(.) is a scalar weighting function of the state at the final time instant N and L(.) is a scalar weighting function of the state vector x_k and the control input vector u_k over the horizon $i \leq k \leq N-1$, both these functions are non linear function. The discrete start time is 'i' and final time is 'N' where$N > i$.

Further the discrete Hamiltonian function is defined, we have

$$H_k = L(x_k \, , u_k) + p_{k+1}^T f(x_k \ , u_k \, , k) \tag{3}$$

Based on equation (1), equation (3) becomes

$$H_k = L(x_k \, , u_k) + p_{k+1}^T f(x_{k+1} - x_k) \tag{4}$$

There exists an optimal solution or optimal control to a problem which can minimize the cost function J_i, if the below conditions are satisfied:

(i) The impulse vector p and the state vector x satisfy the below differential equations:

$$x_{k+1} - x_k = \frac{\partial H_k}{\partial p_k+1} = f(x_k , u_k , k) \quad (5)$$

$$p_{k+1} - p_k = -\frac{\partial H_k}{\partial x_k} \quad (6)$$

The solutions to the above two equations are found based on the known initial value of state vector x and final boundary condition of the impulse vector as mentioned below:

$$x_i = x_0 \quad (7)$$

$$p_N = \frac{\partial S}{\partial x_N} \quad (8)$$

(ii) The Hamiltonian function H_k must have an absolute minimum or maximum (This thesis is dealing with optimal control problem, hence the cost function is subjected to be minimized) with respect to the unknown control $u_k \epsilon U$ where U is the allowed control space. This must hold true for all time instants $k = i \ to \ N-1$, with the constraints on the control vector included. Conditions for such a minimum value is:

$$\frac{\partial H_k}{\partial u_k} = 0 \quad and \quad \frac{\partial^2 H_k}{\partial u_k^2} > 0 \quad (9)$$

From equation (9) the optimal control u_k is obtained and only the first control signal is used for control of the process.

2.1.2 Discrete optimal control of linear systems

Section 2.1.1, describes about certain conditions for non-linear systems to have optimal control solutions. Non-linear systems model are converted to Linearized model to solve it. The following ideas and equations are based on Lecture notes “Optimal model based Control”(Ruscio, 2012c). Consider a process described by the discrete time state space model

$$x_{k+1} = A_k x_k + B_k u_k \quad (10)$$

As this is a common notation which means x_k is a state vector of the dynamic process with$x_k \epsilon R^n$, control vector $u_k \epsilon R^r$, transition matrix $A_k \epsilon R^{nXn}$ and $B_k \epsilon R^{nXr}$ is a control input system matrix.

The next step would be considering a cost function/optimal criterion in a Linear Quadratic form, which is mentioned in equation (11).

$$J_i = \frac{1}{2}\, x_N^T S_N x_N + \frac{1}{2} \sum_{k=i}^{N-1} (x_k^T Q_k x_k + u_k^T P_k u_k) \tag{11}$$

S_N, Q_k and P_k are symmetric matrices, these matrices are time variant, varying in each time step. Based on equation (10), subtract x_k on both sides of equation (10)

$$x_{k+1} - x_k = A_k x_k + B_k u_k - x_k \tag{12}$$

$$x_{k+1} - x_k = (A_k - I) x_k + B_k u_k \tag{13}$$

To find the optimal control u_k^* which minimize the optimal criterion equation (11), below mentioned Hamiltonian equation is written

$$H_k = \frac{1}{2}(x_k^T Q_k x_k + u_k^T P_k u_k) + p_{k+1}^T (x_{k+1} - x_k) \tag{14}$$

Based on equation (13), the equation (14) is modified as

$$H_k = \frac{1}{2}(x_k^T Q_k x_k + u_k^T P_k u_k) + p_{k+1}^T (A_k - I) x_k + B_k u_k \tag{15}$$

Now, the optimal control is given by differentiating equation (15) with u_k and making it equal to zero, the equation (15) now becomes

$$\frac{\partial H_k}{\partial x_k} = 0 \quad \rightarrow \quad P_k u_k + B_k^T p_{k+1} = 0 \tag{16}$$

This leads to an optimal solution as:

$$u_k^* = -p_k^{-1} B_k^T p_{k+1} \tag{17}$$

Substituting equation (17) back into the state space model equation (10)

$$x_{k+1} = A_k x_k - B_k p_k^{-1} B_k^T p_{k+1} \tag{18}$$

Based on the previous chapter of maximum principle equation (6), the impulse vector can be defined as

$$p_{k+1} - p_k = -\frac{\partial H_k}{\partial x_k} = -Q_k x_k - (A_k - I)^T p_{k+1}$$

$$p_{k+1} - p_k == -Q_k x_k - A_K^T p_{k+1} + p_{k+1} \tag{19}$$

Therefore, $p_k = -Q_k x_k - A_K^T p_{k+1}$

Now, equation (18) and (19) forms an autonomous system and equation (20) denotes state vector and the impulse vector at different time instants at the same side of the equality sign.

$$\begin{bmatrix} x_{k+1} \\ p_k \end{bmatrix} = \begin{bmatrix} A_k & B_k p_k^{-1} B_k^T \\ Q_k & A_K^T \end{bmatrix} \begin{bmatrix} x_k \\ p_{k+1} \end{bmatrix} \tag{20}$$

2.1.3 Discrete time Algebraic Riccati equation

Riccati equations are named after Count Riccati (Ruscio, 2012c). Algebraic Riccati equations can be divided into continues time and discrete time Riccati equations (Lancaster and Rodman, 1995). As the thesis is focused on discrete time system, this chapter presents topics in detail regarding discrete time algebraic Riccati equations.

The discrete time Riccati equation in LQOC solution may be formulated in different ways. The two different formulations are(Ruscio, 2012c):

(i) This equation must hold for an arbitrarily state vector $x_k \neq 0$ and also assumes that the control weighting matrix p_k is non-singular. This gives the following matrix equations for finding R_k

$$R_k = Q_k + A_K^T R_{k+1} (I + B_k P_K^{-1} B_K^T R_{k+1})^{-1} A_k \tag{21}$$

(ii) An alternative formulation in the case when R_{k+1} is non-singular is

$$R_k = Q_k + A_K^T (R_{k+1}^{-1} + B_k P_K^{-1} B_K^T)^{-1} A_k \tag{22}$$

Now in this section two different formulations of discrete time Riccati equation are formulated which does not involve the inversion of the weighting matrix p_k. The below mentioned formulations are most used formulations.

Assume that

$$p_k = R_k x_k \tag{22a}$$

The main aim is to show that there is a linear relationship between the impulse vector p_k and the state vector x_k. This means if there is an equation for defining $R_k$then it proves that there exists such a relationship as described above in equation (22a).

Based on equation (19), it's known that $p_k = Q_k x_k + A_K^T p_{k+1}$, substituting equation (19) into (22a)

$$R_k x_k = Q_k x_k + A_K^T p_{k+1} \tag{23}$$

Based on equation (22), it's assumed$p_{k+1} = R_{k+1} x_{k+1}$, substituting this into equation (23)

$$R_k x_k = Q_k x_k + A_K^T R_{k+1} x_{k+1} \tag{24}$$

An expression for closed loop system is obtained by putting the optimal control $u_k^* = G_k x_k$ and $G_k = -(P_k + B_K^T R_{k+1} B_k)^{-1} B_K^T R_{k+1} A_k$ (this part will be discussed in next section) into the state equation $x_{k+1} = A_k x_k + B_k u_k$, this gives

$$x_{k+1} = (A_k - B_k(P_k + B_K^T R_{k+1} B_k)^{-1} B_K^T R_{k+1} A_k) x_k \tag{25}$$

Now, putting equation (25) into (24), gives

$$R_k x_k = Q_k x_k + A_K^T R_{k+1}(A_k - B_k(P_k + B_K^T R_{k+1} B_k)^{-1} B_K^T R_{k+1} A_k) x_k \tag{26}$$

This equation must hold for all states $x_k \neq 0$, which gives

$$R_k = Q_k + A_K^T (R_{k+1} - R_{k+1} B_k (P_k + B_K^T R_{k+1} B_k)^{-1} B_K^T R_{k+1}) A_k \tag{27}$$

This formulation of the discrete time Riccati equations is always preferred and the only matrix $P_k + B_K^T R_{k+1} B_k$ needs to be inverted. The Boundary condition always remains same as $R_N = S_N$, where S_N is the weighting matrix for the final state x_N.

The fourth formulation of the Riccati equation is presented as:

$$R_k = (A_k + B_k G_k)^T R_{k+1}(A_k + B_k G_k) + G_K^T P_k G_k + Q_k \tag{28}$$

$$G_k = -(P_k + B_K^T R_{k+1} B_k)^{-1} B_K^T R_{k+1} A_k \tag{29}$$

This formulation of the discrete time Riccati equation is known as Joseph's stable version of the Riccati equation. This equation consists only of symmetric terms, and this formulation is more preferred in numerical calculations.

It's also observable that for a given control gain matrix G_k, the equation (28) is a discrete time Lyapunov equation. Equation (28) and (29) with advantage be used in order to iterate to find the stationary solution to the LQOC problem with infinite horizon.

2.2 Integral Action in Discrete LQ Optimal Controller

Integral action is required in Discrete LQOC so that output should follow the setpoint in the best way or zero steady state error is to be observed. The theory and equations described below are based on journal paper "Discrete LQ Optimal Control Integral Action" (Ruscio, 2012a).

Let's consider a process model as described below with unknown slowly varying noises:

$$x_{k+1} = Ax_k + Bu_k + v \tag{30}$$

$$y_k = Dx_k + w \tag{31}$$

where $k \geq i$ is the discrete time and initial state x_i is given and $x_k \,\epsilon\, R^n$ is the state vector, control vector $u_k \,\epsilon\, R^r$, output measurement vector $y_k \,\epsilon\, R^m$ and A,B and D are known system matrices of appropriate dimensions. The disturbance v and w are both unknown, i.e. v is an unknown constant or a slowly varying process disturbance and w is an unknown constant or a slowly varying measurement noise vector.

Point to be noted is that the model equation (30) and (31) may arise from linearizing non linear models around some nominal steady state and input variables or from system identification.

The main aim of this controller is to make the output y_k to be as close as possible to a known reference vector r_k. In this case, it makes sense to use a control input u_k which minimizes a control objective where the deviation $r_k - y_k$ is weighted in the objective, but control action costs, so the control input u_k is also weighted in the objective.

The Control Objective/Performance index/Cost function for large or infinite prediction horizon N

$$J_i = \frac{1}{2}\sum_{k=i}^{\infty}((y_k - r_k)^T Q_k (y_k - r_k) + \Delta u_k^T P_k \Delta u_k) \tag{32}$$

Where $Q_k \epsilon R^{mXm}$ and $P_k \epsilon R^{rXr}$ is symmetric weighting matrices. The reference vector r is treated as constant or slowly varying in the design phase of the LQ optimal controller with integral action for MIMO system. Assume P>0.

Based on the state equation of (30) and (31), we can re write into

$$x_k = Ax_{k-1} + Bu_{k-1} + v \tag{33}$$

$$y_{k-1} = Dx_{k-1} + w \tag{34}$$

Subtracting equation (30)-(33) and (31)-(33)

$$x_{k+1} - x_k = Ax_k - Ax_{k-1} + Bu_k - Bu_{k-1} + v - v$$

$$\Delta x_{k+1} = A\Delta x_k + B\Delta u_k \tag{35}$$

Similarly

$$y_k - y_{k-1} = Dx_k - Dx_{k-1} + w - w$$

$$y_k = y_{k-1} + D\Delta x_k \tag{36}$$

Now augmented matrix (the matrix is made greater in size) can be constructed from the latest formulations and is expressed as

$$\begin{bmatrix} x_{k+1} \\ y_k \end{bmatrix} = \begin{bmatrix} A & 0_{nXm} \\ D & I_{mXm} \end{bmatrix} \begin{bmatrix} \Delta x_k \\ y_{k-1} \end{bmatrix} + \begin{bmatrix} B \\ 0_{mXr} \end{bmatrix} \Delta u_k \tag{37}$$

$$y_k = [D \quad I_{mXm}] \begin{bmatrix} \Delta x_k \\ y_{k-1} \end{bmatrix} \tag{38}$$

The performance index in equation (32) with r=0 and the augmented state space model in (37) and (38) define a standard LQ control problem. If r is a non-zero constant reference then the measurement equations (37) can be written as

$$y_k - r = y_{k-1} - r + D\Delta x_k \tag{39}$$

Hence, the state and output equation (37) and (38) can be rewritten as:

$$\begin{bmatrix} x_{k+1} \\ y_k - r \end{bmatrix} = \begin{bmatrix} A & 0_{nXm} \\ D & I_{mXm} \end{bmatrix} \begin{bmatrix} \Delta x_k \\ y_{k-1} - r \end{bmatrix} + \begin{bmatrix} B \\ 0_{mXr} \end{bmatrix} \Delta u_k \tag{40}$$

$$y_k - r = [D \quad I_{mXm}] \begin{bmatrix} \Delta x_k \\ y_{k-1} - r \end{bmatrix} \tag{41}$$

The above equation can be rewritten shortly as

$$\tilde{x}_{\cdot k+1} = \tilde{A}\,\tilde{x}_{\cdot k} + \tilde{B}\Delta u_k \tag{42}$$

$$\tilde{y}_{\cdot k} = \widetilde{D}\,\tilde{x}_{\cdot k} \tag{43}$$

The pair $(\tilde{A}, \tilde{B})$ is stabilizable. Hereafter, Hamilton matrix defined and relation between impulse vector and state vector is assumed (refer above section for detail discussion). The solution to the LQ optimal control minimizing the performance index (32) with respect to the control deviation Δu_k subject to the state equation (41) and (42) is given by the state feedback

$$\Delta u_k = G\tilde{x}_{\cdot k} \tag{44}$$

Where the feedback matrix G is obtained as

$$G = -(P + \widetilde{B^T}\, R\, \tilde{B})^{-1}\, \widetilde{B^T}\, R\tilde{A} \tag{45}$$

Where R is the positive solution to the discrete solution to the discrete time algebraic Riccati equation

$$\begin{aligned} R &= \tilde{Q} + \widetilde{A^T} R\tilde{A} - \widetilde{A^T} R\tilde{B}(P + \widetilde{B^T} R\tilde{B})^{-1} \widetilde{B^T} R\tilde{A} \\ &= \tilde{Q} + G^T P G + (\tilde{A} + \tilde{B}G)^T R(\tilde{A} + \tilde{B}G) \end{aligned} \tag{46}$$

Where the above formulation of the Riccati equation is known as the Joseph's stable version which ensures symmetry of the solution R. Based on equation (44) the controller on incremental form is:

$$\Delta u_k = [G_1\; G_2] \begin{bmatrix} \Delta x_k \\ y_{k-1} - r \end{bmatrix} \tag{47}$$

The above solution is not directly used in calculation, rather using the known equation$u_k = u_{k-1} + \Delta u_k$, the above equation (47) is modified as

$$u_k = u_{k-1} + G_1 \Delta x_k + G_2(y_{k-1} - r_k) \tag{48}$$

The above equation is the resulting control input signal for the state space equation.

3. Model Predictive Control

Tradition feedback controllers (PID Controllers) were most dominated control strategy used in industrial process. These controllers operate by adjusting control action in response to a change in the setpoint of a system. Hence a more efficient/optimal control strategy was required due to growing quality requirement and cost management. Focus was made on advance control/predictive control. Model predictive control is a technique that focuses on constructing controllers that can adjust control action before a change in the setpoint actually occurs.

Evolution of Model Predictive Controller(Morari, 2008):

- 1980: Seminar By Haydel and Prett at U.Wisconsin on work with cutler and Ramaker
- Early 1980s: Work with Garcia on Internal Model Control
- 1993: rawlings & Muske, Stability of Receding Horizon Control. IEEE-TAC
- 2000: Mayne, Rawlings,Rao,Scokaert: MPC, Stability & Optimality. Auomatica
- 2003: Qin & Badwell: Survey of Industrial MPC Technology Control Eng Practice.

Several Authors have published excellent reviews of MPC theoretical issues including the paper of Garcia et al. (Garcia et al., 1989), Ricker (Ricker, 1991)and Rawlings(Rawlings et al., 1994). With over 2000 industrial installation model predictive control is currently most widely implemented advance process control technology for process plant(Nikolaou, 2011).

Model Predictive control (MPC) refers to a class of computer control algorithm that utilize an explicit process model to predict the future response of a plant. At each control interval, an MPC algorithm attempts to optimize future plant behaviour by computing a sequence of future manipulated variable adjustment. The first input in the optimal sequence is then sent into the plant, and the entire calculation is repeated at subsequent control intervals. Programming tools like MATLAB (Matrix Laboratory, Model Predictive Control Toolbox) and Labview (Laboratory Virtual Instrumentation Engineering Workbench, Control Design and Simulation Module) has MPC functionality (Halvorsen., 2011).

Further part of this chapter describes about introduction to MPC in 3.1. Subsection 3.1.1 provides various definitions of critical terms, subsection 3.1.2 provides the theoretical description of MPC and section 3.2 provides the equations for MPC with integral action.

3.1 Introduction

Model Predictive Control refers to a class of algorithm that compute a sequence of algorithm that computes a sequence of manipulated variable adjustment in order to optimize the future behaviour of a plant. Originally developed to meet the specialized control needs of power plant and petroleum refineries (Qin and Badgwell). Model predictive control is a control strategy which is a special case of the optimal control theory developed in 1960 and later (Ruscio, 2012b). Receding Horizon control, also known as model predictive control (Nicola et al., 2000).

Model Predictive control can be divided into

- Non-Linear Model Predictive Control
- Linear Model Predictive Control

The above classification is based on the linear or non-linear model used in the process and prediction model. Most of the MPC used in daily life are linear as non-linear model predictive control is not guaranteed to converge in fixed computation time(Ruscio, 2012b).

3.1.1 Definitions

MPC algorithm consist of various terms and their definitions are as follows (Ruscio, 2012b)

- ***Prediction Horizon***
  The symbol 'L' denotes prediction horizon, which means the number of samples in the future the MPC controller predicts the plant output (Halvorsen., 2011).

- ***Control Horizon***
  The number of samples within the prediction horizon where the MPC controller can affect the control action (Halvorsen., 2011)

- ***Output Error Weightings (Q)***
  Specifics the weight matrix Q for each system output error in cost function. The dimension of this matrix must match the number of plant outputs (National, 2012). It is symmetric and positive semi-definite weighting matrices specified by user. The more specific choice would be a diagonal weighting matrices. The weighting matrices are almost always chosen as time invariant matrices such that the weighting matrices are constant over the prediction horizon L so that $Q_1 = Q_2 = ... = Q_L$.

- ***Control Action change weightings (R)***
  Specifies the weight matrix R for each rate of control action change in cost function. The dimension of this matrix must match the number of plant inputs (National, 2012). It is symmetric and positive semi-definite weighting matrices specified by user. The more specific choice would be a diagonal weighting matrices. The

weighting matrices are almost always chosen as time invariant matrices such that the weighting matrices are constant over the prediction horizon L so that R_1 = R_2 =.....= R_L.

- ***Control Action Error Weightings (P)***

  Specifies the weight matrix P for each control action error change in cost function and recommends to specify this matrix only for system with more inputs than outputs (National, 2012). Its symmetric and positive semi-definite weighting matrices specified by user. Often, P matrix is chosen as zero in order to obtain MPC with offset free control i.e. y=r in steady state.

- ***Cost Function***

  The Control Objective/Cost function J_k , which is a scalar criterion measuring the difference between future output $y_{k+1/L}$ and specified reference (future) $r_{k+1/L}$ and at the same time recognizing that the control u_k is costly. The objective is a measure of the process behaviour over the prediction horizon L. This objective is minimized with respect to the future control vectors $u_{k+1/L}$ and optimization process is solved again at the next time instant K=K+1.

The common control objective used is given by scalar function

$$J_k = \sum_{i=1}^{L}((y_{k+i} - r_{k+i})^T Q_i (y_{k+i} - r_{k+i}) + u_{k+i-1}^T P_i u_{k+i-1} + \Delta u_{k+i-1}^T R_i \Delta u_{k+i-1}) \quad (49)$$

The summation loop runs from time instants one and until it reaches the Prediction horizon (L). $Q_i \epsilon R^{mXm}$, $P_i \epsilon R^{rXr}$ and $R_i \epsilon R^{rXr}$ are weighting matrix as described above. The problem of choosing these matrices are usually process dependent and must usually be trial and error.

The matrix formulation of the objective J_k will be

$$J_k = (y_{k+1/L} - r_{k+1/L})^T Q (y_{k+1/L} - r_{k+1/L}) + u_{k/L}^T P_i u_{k/L} + \Delta u_{k/L}^T R_i \Delta u_{k/L}) \quad (50)$$

Where$Q \epsilon R^{LmXLm}$, $P \epsilon R^{LrXLr}$ and $R \epsilon R^{LrXLr}$ are symmetric and positive semi definite block diagonal weighting matrices.

Where 'K' is the starting point of the matrix and 'L' is the ending row of the matrix. For example: Let's take L=4.

$$\overbrace{\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ y_{k+3} \\ y_{k+4} \end{bmatrix}}^{y_{k+1|4}} \quad \text{Or} \quad \overbrace{\begin{bmatrix} u_k \\ u_{k+1} \\ u_{k+2} \\ u_{k+3} \end{bmatrix}}^{u_{k|4}} \quad \text{Or} \quad \underbrace{\begin{bmatrix} y_{k+1} - r_{k+1} \\ y_{k+2} - r_{k+2} \\ y_{k+3} - r_{k+3} \\ y_{k+4} - r_{k+4} \end{bmatrix}}_{(y_{k+1|4} - r_{k+1|4})}$$

The control problem is subjected to minimize the cost function.

- ***Prediction Model:***

  The linear dynamic process model can always be written as a prediction model (PM) which takes the standard form

  $$y_{k+1|L} = p_L + F_L u_{k|L} \quad (51)$$

  Where $F_L \in R^{LmXLr}$ a constant matrix is derived from the process model and $p_L \in R^{Lm}$ is a vector which is in general is dependent of a number of inputs and outputs older than time K as well as the model parameters. Equation (51) can be used directly in MPC algorithm which is computing the actual control input vector.

  Some algorithm for MPC are computing process deviation variable such that computing the vector $\Delta u_{k|L}$ of future control deviation variables. Then $u_k = \Delta u_k + u_{k-1}$ is used as the actual control vector, for this case the prediction model can be written as $y_{k+1|L} = p_L^{\Delta} + F_L^{\Delta} u_{k|L}$.

- ***Constraints:***

  Constraints are a sought of limitation or boundary value given to a variable. Constraints can be applied to many variables. These are discussed in detail in chapter 4.

3.1.2 Theory

MPC is a predictive controller which will predict the future control signal of the system/plant process. This section describes this theory in detail.

MPC consist of an optimization problem at each time instant k. The main point of this optimization problem is to compute a new control input vectoru_k, to be feed to the system and at the same time take process constraints into consideration. An MPC is a computer based algorithm and it consists of(Halvorsen., 2011):

- Model of the Process
- Cost Function
- Constraints

According to Holkar and Waghmare (Holkar. and L.M.Waghmare, 2010) MPC usually contains the following three ideas:

- Explicit use of a Model to predict the process output along a future horizon.
- Calculation of a control sequence to optimize a performance index
- A receding horizon strategy, so that at each time instance the horizon is moved towards the future, which involved the application of the first control signal of the sequence calculated at each step.

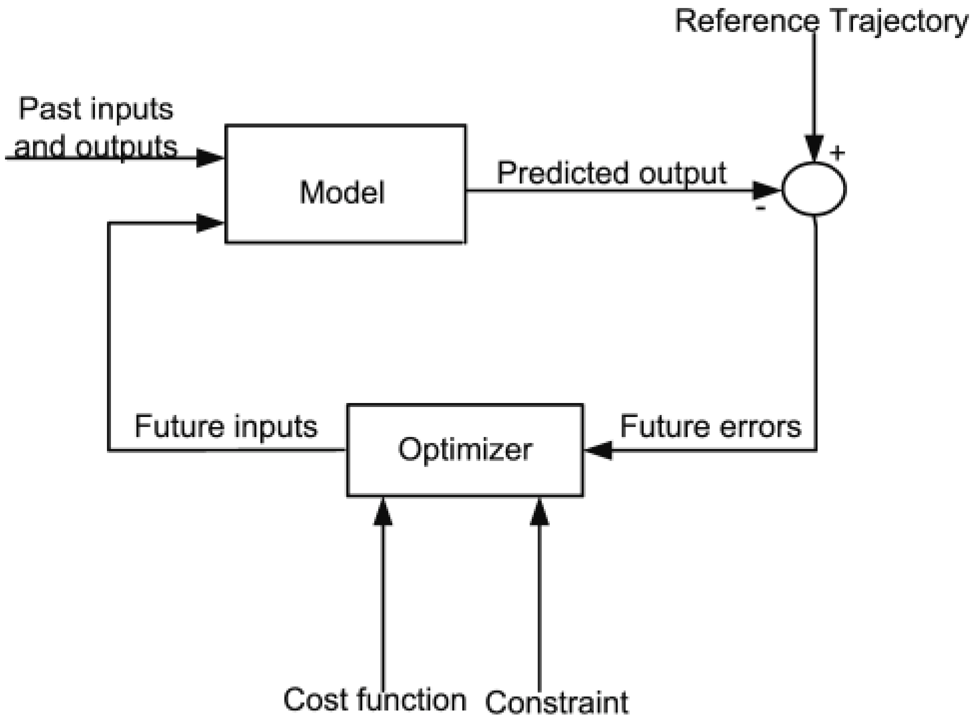


Figure 3-1: *Sketch of an MPC Controller(Halvorsen., 2011)*

Refer figure 3-1, the sketch of Model Predictive controller and according to Hans-Petter Halvorsen (Halvorsen., 2011), Model predictive controller refers to a class of computer control algorithm that utilize an explicit process model to predict the future response of a plant. At each control interval an MPC algorithm attempts to optimize the future plan behaviour by computing a sequence of future manipulated variable adjustment. The first input in the optimal sequence is then sent into the plant and the entire calculation is repeated at subsequent intervals.

The above theoretical concept is provided in a pictorial representation as a moving horizon MPC. Refer figure 3-2, where x axis represents time and y axis represents the value of control and output measurements. For the current time instant ‘t’, the prediction horizon would be ‘t+N’ and the control horizon will be less than the predication horizon.

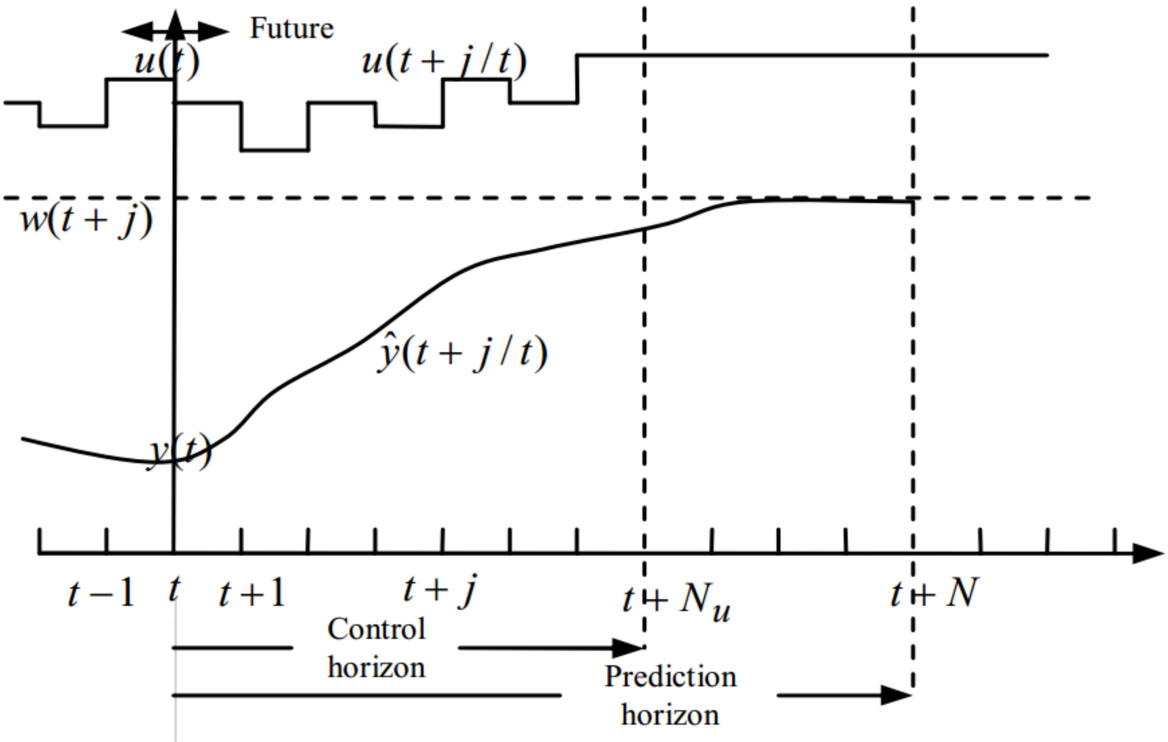


Figure 3-2: *The moving horizon strategy of MPC (Holkar. and L.M.Waghmare, 2010)*

The MPC methodology is characterized by the strategy as represented in figure 3-2 (Holkar. and L.M.Waghmare, 2010)

1) The process model calculates the predicted future output for the prediction horizon (N) at each time instant t. These depends upon the known values up to instance t (past inputs and outputs) including the current output (initial condition of y (t)) and on the future control signals to be calculated.
2) The sequence of future control signals is computed to optimize a performance index. Usually the control effort is included in the performance index.
3) Only the current control signal u(t) is send to process, at the next sampling instant y(t+1) is measured and step 1 is repeated and all sequence brought up to date.

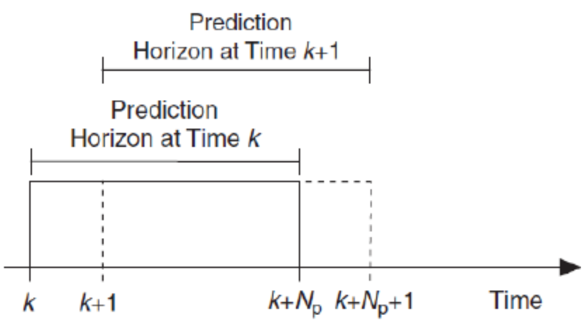


Figure 3-3, represents the moving Prediction horizon for next instant of time. Hence the time instant increases by k+1 and prediction horizon increases by k+N+1.

Figure 3-3: *Moving horizon representation for MPC (Halvorsen., 2011).*

With reference to figure 3-4, the above concept is provided in a flowchart or the critical for loop is provided in a pictorial represents in Nikolaou (Nikolaou, 2011). As the concept is discussed in detail above, hence this picture is not explained in detail as its self explanatory.

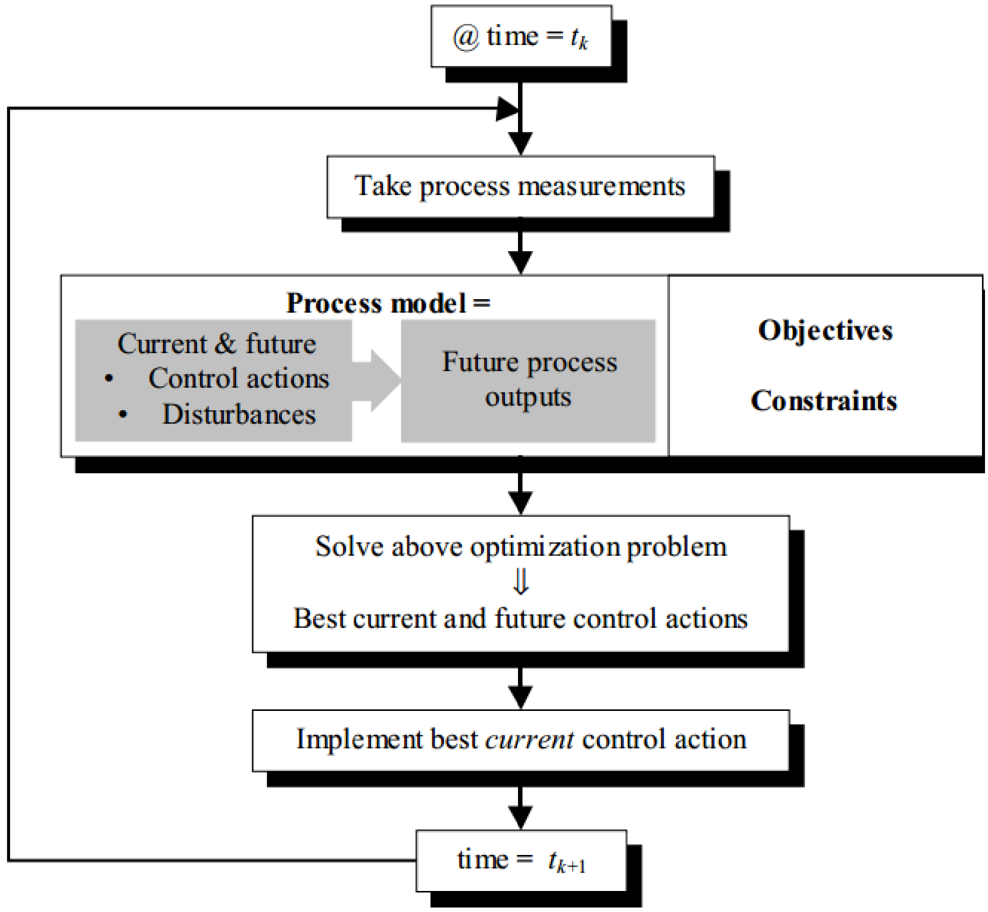


Figure 3-4: *Model Predictive Control Scheme (Nikolaou, 2011).*

3.2 Model Predictive control with Integral Action

MPC with integral action uses control deviation variable $\Delta u_{k/L}$ to obtain offset free control and with Δu_k gives offset free control if the weighting matrix P=0. Another Advantage of computing $\Delta u_{k/L}$ and choosing P=0 is to avoid practical problems with non zero mean variables (Ruscio, 2012b).

The below discussed equation and theory are based on (Ruscio, 2012b). A proper deterministic linear dynamic system can be written as a state space model

$$x_{k+1} = Ax_k + Bu_k \tag{3.1}$$

$$y_k = Dx_k \tag{3.2}$$

The above state space model is converted to augmented or extended state space model as described in section 2.2.

The matrix equivalent of cost function as discussed in detail in section 3.1.1. Note: this cost function consists of deviation variable $\Delta u_{k/L}$.

$$J_k = (y_{k+1/L} - r_{k+1/L})^T Q (y_{k+1/L} - r_{k+1/L}) + u_{k/L}^T P_i u_{k/L} + \Delta u_{k/L}^T R_i \Delta u_{k/L}) \tag{3.3}$$

A prediction model in terms of process deviation can be derived from $y_{k+1|L} = p_L + F_L\, u_{k|L}$ and $u_{k|L} = S\Delta u_{k|L} + Cu_{k-1|L}$ this is build from state space equation in 3.1 and 3.2 and later based on $u_k = \Delta u_k + u_{k-1}$, for detail derivation refer to (Ruscio, 2012b). The matrices S and c consist of ones and zero, Hence we have prediction as

$$y_{k+1|L} = F_L{}^{\Delta} \Delta u_{k|L} + P_L{}^{\Delta} \tag{3.4}$$

Where

$$F_L{}^{\Delta} = F_L S \tag{3.5}$$

$$P_L{}^{\Delta} = P_L + F_L C u_{k-1} \tag{3.6}$$

Where $\mathrm{p_L} = \mathrm{O_L A x_k}$ and $\mathrm{F_L} = [\mathrm{O_L B} \quad \mathrm{H_L^d}]$, $\mathrm{O_L}$ is the extended observability matrix of and $\mathrm{H_L^d}$ is the toeplitz matrix.

Now, substituting (3.4) into (3.3) with P=0 for achieving integral action:

$$J = (F_L{}^{\Delta} \Delta u_{k|L} + P_L{}^{\Delta} - r_{k+1/L})^T Q (F_L{}^{\Delta} \Delta u_{k|L} + P_L{}^{\Delta} - r_{k+1/L}) + \Delta u^T{}_{k/L} R \Delta u_{k/L}$$

$$J_k = (F\Delta U + P - r)^T Q (F\Delta U + P - r) + \ \Delta U^T P \Delta U$$

$$J_k = \Delta U^T F^T Q F \Delta U + \Delta \underbrace{U^T F^T Q(P-r) + (P-r)^T Q \Delta U F + \Delta U^T P \Delta U}_{2(P-r)^T QUF} + \underbrace{(P-r)^T Q(P-r)}_{J_o}$$

$$J_k = \Delta U^T (F^T Q F + P) \Delta U + 2(P-r)^T Q \Delta U F + J_o$$

Gives,

$$J_k = \Delta U_{K/L}^T H \Delta U_{K/L} + 2 f_K^T \Delta U_{K/L} + J_o \tag{3.5}$$

Where= $F_L^{\Delta T} Q F_L^{\Delta} + R + S^T P S$, $J_o = (p_L^{\Delta} - r_{k+1/L})^T Q \left(p_L^{\Delta} - r_{k+1/L}\right) + U_{K-1}^T c^T P\ c\ U_{k-1}$ and $f = F_L^{\Delta T}\ Q(p_L^{\Delta} - r_{k+1/L}) + S^T P c U_{k-1}$

Now, minimizing the control objective in equation (3.5) with respect to $\Delta U_{K/L}$ and equating it to zero which gives

$$\frac{dJ_k}{d\Delta U_{K/L}} = 0 = \frac{= d(\Delta U_{K/L}^T H \Delta U_{K/L} + 2 f_K^T \Delta U_{K/L} + J_o)}{d\Delta U_{K/L}}$$

$$\frac{dJ_k}{d\Delta U_{K/L}} = 2H\Delta U_{K/L} + 2f = 0$$

$$\Delta u_{K/L}^* = -H^{-1} f \tag{3.6}$$

The above is also known as unconstrained MPC controller, where $\Delta u_{K/L}^*$ is a vector of length 'L', which consist of the future deviational control signal for the system. As we know $u_k = \Delta u_k + u_{k-1}$, hence only the first value from $\Delta u_{K/L}^*$ is used to find the control signal u_k.

According to (Ruscio, 2012b), Advantage of using integral action is zero steady state error and the resulting controller is insensitive to non-zero mean control variables and constant disturbance. Most importantly, it leads to an MPC which are computing control deviation variable $\Delta u_{k/L}$.

4. Handling Constraints

Today’s process need to be operated under tighter performance specification and at the same time more and more constraints, stemming for example from environmental and safety consideration need to be satisfied (Findeisen. and Allgower). According to Nunes (Nunes, 2001), all process are subjected to restrictions and these can be considered in the objective function as constraints in input and outputs. The main requirements for constraints are mostly because badly tuned optimal controllers tend to take the system/plant to extreme and sometimes unstable condition like going beyond the physical limit of valve opening.

According to authors (Lee. et al.) to have a stabilizing receding horizon control is to adopt a finite input and state horizon with the *terminal equality constraint* that all states should be zero within a finite horizon. Another approached provided in this paper is to introduce *invariant ellipsoid constraint* in order to relax terminal equality constraints.

Constraints are an important part of this thesis and it’s discussed in detail in this chapter. This chapter is divided into various sections where section 4.1 describes about classification of constraints, section 4.2 provides the different type of constraints, section 4.3 provides the equation and theory for handling constraints in MPC and section 4.4 provides the equation and theory for handling constraints in LQ optimal control.

4.1 Classification of Constraints

Constraints can be classified into different types and they are shown below.

Hard Constraints:

They are inequality constraints (e.g. lower and upper bounds) on parameters to be estimated (Benavoli et al.). Hard constraints are those which definitely need to be satisfied.

Soft Constraints:

These constraints would like to true but not at the expense of the hard constraints example cost function are called soft constraints (Encylopedia, 2012).

Linear Equality Constraints:

Linear equality constraints take the form $Ax = b$, where A and b has m rows. Note this is usually an underdetermined system. Most optimization do not operate this way because of numerical errors in computing N (Hauser, 2012).

Linear Inequality Constraints:

Most linear MPC methods lead to a quadratic problem with inequality constraints. They take the form $Ax \leq b$ which is solved iteratively. The numbers of iterations are finite and the solution is guaranteed to converge if the Quadratic problem is well formulated and a solution is feasible. Where $A \in R^{mXn}$ and $b \in R^{m}$. Note that this case can allow the number of inequality constraints m to be greater than the number of parameter n in x (Ruscio, 2012b).

4.2 Types of Constraints

Different type of constraints exists in order to obtain a better performance from optimal controller. For simplicity single input single output constraints are discussed. The following ideas are based on (Ruscio, 2012b)

- ***Input Amplitude / Control Variable Amplitude Constraint***
  These are the most common encountered constraints among all constraints. These are physical hard constraints on the system. For example

  $$u^{min} \leq u(k) \leq u^{max} \tag{4.1}$$

  A typical example for these type of constraints are valve which cannot operate beyond 100% open or a certain voltage or current valve to go beyond a certain range.

- ***Input rate/Control Variable Incremental Constraint***
  These are hard constraints are rate of change in control signal constraints, for some process it's not advisable to have a huge increase in the control variable. Hence limiting the rate of change will in turn limit the control signal. It's denoted by
  $$\Delta u^{min} \leq \Delta u(k) \leq \Delta u^{max} \tag{4.2}$$

- ***Output Constraints***
  These constraints are typically operating range for the plant output. For instance the output y(k) has an upper limit y^{max} and a lower limit y^{min}, then the output constraints are specified as

  $$y^{min} \leq y(k) \leq y^{max} \tag{4.3}$$

- ***State Constraints***
  These constraints are applied on state variable, if they are measurable or they posses critical outcome. (Wang, 2009)

4.3 Constraints Handling in MPC

The constraints in MPC can be handled in two types of programming

1) Algorithm based constraints, which is discussed in this section in detail
2) Unconstrained MPC can be limited with an if else loop constraints, if else loop is discussed in detail in section 4.4 of LQ optimal control.

MPC uses *Deviation variable* and *linear inequality constraints*. The following equations are based on Ruscio (2012 b).

Given a linear quadratic objective function which is based on deviation variable

$$J_k = (y_{k+1/L} - r_{k+1/L})^T Q (y_{k+1/L} - r_{k+1/L}) + \Delta u_{k/L}^T P_i \Delta u_{k/L} + u_{k/L}^T R_i u_{k/L})$$

Minimizing J with respect to the future control inputs, subject to input and output constraints, this problem can be formulated as follows:

$$\min_{\Delta u_{k/L}} J$$

Subject to

$$u_{k/L}^{min} \leq \quad u_{k/L} \quad \leq u_{k/L}^{max} \qquad \text{(Input amplitude constraints)}$$

$$\Delta u_{k/L}^{min} \leq \Delta u_{k/L} \leq \Delta u_{k/L}^{max} \qquad \text{(Input change constraints)}$$

$$y_{k+1/L}^{min} \leq y_{k+1/L} \leq y_{k+1/L}^{max} \qquad \text{(Output constraints)}$$

The above can be written as a linear inequality as:

$$A \Delta u_{k/L} \leq b$$

Where

$$A = \begin{bmatrix} S \\ -S \\ I \\ -I \\ F_L \\ -F_L \end{bmatrix}, \; b = \begin{bmatrix} u_{k/L}^{max} - c u_{k-1} \\ u_{k/L}^{min} - c u_{k-1} \\ \Delta u_{k/L}^{max} \\ -\Delta u_{k/L}^{min} \\ y_{k+1/L}^{max} - p_L(k) \\ -y_{k+1/L}^{min} - p_L(k) \end{bmatrix}$$

Where $p_L(k)$ is defined in terms of known past inputs and outputs, F_L is a constant lower triangular matrix. Where matrices $S \; and \; I$ found to statisfes the following the relationship

$$u_{k/L} = S \Delta u_{k/L} + c u_{k-1}$$

Matrices $S \; and \; I$ are found through a Matlab program 'scmat.m' written by Rusico (2012,b). These matrices contain ones and zeros only.

The above quadratic problem with constraints can be solved in Matlab by the optimization toolbox function as:

$$\Delta u_{k/L} = qp(H, f, A, b)$$

4.4 Constraints Handling in LQ Optimal Control

MPC handles constraints in its algorithm whereas LQ optimal control doesn't handle in its algorithm. But still hard limit constraint can be used in the form of if-else loop. An example for Input change constraints $\Delta u_{k/L}$ is shown. Let the constraint be $\Delta u_{k/L}^{min} \leq \Delta u_{k/L} \leq \Delta u_{k/L}^{max}$, where $\Delta u_{k/L}^{min}$ *and* $\Delta u_{k/L}^{max}$, are minimum and maximum value for input change is. It's known that

$$\Delta u_k = G_1 \Delta x_k + G_2 (y_{k-1} - r_k)$$

The if-else loop can be implemented as:

```
if Δu_k (1,1) > Δu_{k/L}^{max}
    Δu_k (1,1) = Δu_{k/L}^{max};
elseif Δu_k (1,1) < Δu_{k/L}^{min}
    Δu_k (1,1 Δu_{k/L}^{min};
end
```

The above algorithm can be used for Input amplitude constraints and Output constraints and also if there are many inputs or outputs the same loop algorithm can be used but with different index like Δu_k `(1,2)`.

5. Kalman Filter

The four tank process (which will be discussed chapter 7) used for simulation has state variable measurement only in two tanks, other two tanks level are not measured. There is a need to estimate the level or state variable estimation, Kalman Filter is used to estimate the level of the two tanks. If States are not measured then x_k may be computed from the knowledge of a number of past inputs and outputs over the past horizon J. The states can also be estimated in a state observe, using Kalman filter gain (Ruscio, 2012b).

The Kalman filter is a set of mathematical equation that provides an efficient computational (recursive) solution of the least square methods (Welch and Bishop). The Kalman filter is a state estimator which produces an optimal estimate in the sense that the mean value of the sum of the estimation error gets a minimal value. Kalman filter is a set of mathematical algorithm which is implemented in the simulation. The algorithm is described briefly below.

Let's consider a process in state space form as $x_{k+1} = Ax_k + Bu_k + v_k$ and $y_k = Dx_k + w_k$ Where v_k and w_k are white process and measurement noise respectively. The Kalman filter K is a nXr matrix, where n is the number of states need to be observed. The Kalman filter gain is found in simulation using "dlqe", given the covariance of the noises, a Kalman gain K can be calculated using the function dlqe (Verhaegen et al., 2007). Set the initial state estimate as: $\overline{x_0} = x_o$

Measurement Estimate: $\bar{y}(k) = \overline{x_1}(k)$

Estimator Error: $e(k) = y(k) - \bar{y}(k)$

Corrected State Estimate:

$$\widehat{x_1}(k) = \overline{x_1}(k) + K_1 e(k)$$
$$\widehat{x_2}(k) = \overline{x_2}(k) + K_2 e(k)$$

Predicted State Estimate:

$$\widehat{x_1}(k+1) = \widehat{x_1}(k) + T_S k_P U(k) - T_S \widehat{x_2}(k)$$
$$\widehat{x_2}(k+1) = \widehat{x_2}(k)$$

The above equation are based on (Halvorsen., 2011). These 4 steps are running inside a loop and calculate new estimates for each iteration of the MATLAB code. Note the different notation may be used:

Apriori (Predicted) estimate: $x_p = \bar{x}$ and Aposteriori (Corrected) estimate: $x_c = \hat{x}$

6. PID

Proportional Integral Derivative (PID) is the most commonly used controller in the industries as it doesn't require the model of the process. There are many forms of the PID controllers, but discrete type of PID controllers are more preferred as they easy to implement in computer software and for the simulation velocity form of the PID controllers used because it doesn't require specification of the bias term $\bar{p}$ and is less prone to reset windup(Triratanajaru, 2011).

Refer below for the velocity form of PI Controller and for complete derivation refer to (Ruscio, 1996),

$$u_k = u_{k-1} + G_1(y_k - y_{k-1}) + G_2(y_{k-1} - r)$$

Where

$$G_1 = -k_p$$

$$G_2 = -\frac{K_p}{T_i}\Delta t$$

The above discussion shows that the PI controller is exactly of the same structure as a state feedback controller (LQ optimal Controller). The controller takes feedback from the deviation state vector $\Delta x_k = x_k - x_{k-1}$ while the PI controller only uses feedback from the output deviation $\Delta y_k = D\Delta x_k$ (Ruscio, 1996).

Two separate PI controllers are used to control the tank 1 and tank 2, the values of P and I are chosen after trial and error method. The value of PI for minimum phase system is kp = [8 12]; Ti= [10 4] and the value for non minimum phase system is kp = [15 -.12]; Ti = [100 250].

6.1 RGA Analysis

As the quadruple tank is a MIMO system, hence pairing is one of the main concepts to analysis before implementing the PI controller. The following equation are based on (Ruscio, 1996). The quadruple tank has 2 inputs, hence lets discuss an example of 2x2 system where u_1 and u_2 are input and $y_1 and\ y_2$ are output variables. The system can be represented as

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

The above equation can be written as:

$$y = H_p u$$

The relative gain array is given by

$\Lambda = H_p\ (H_p^{-1})^T$ (Element by element multiplication), where H_p is a non singular matrix and Λ is described by

$$\Lambda = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix} = \begin{bmatrix} \lambda_{11} & 1-\lambda_{11} \\ 1-\lambda_{11} & \lambda_{11} \end{bmatrix}$$

Where $\lambda_{11} = \frac{1}{1-\frac{x_{12}\,x_{21}}{x_{11}x_{22}}}$, The pairing of the variables depends on

1. Chose the pairing $x_i \rightarrow y_i$, for which the corresponding RGA element λ_{ij} is positive and so close to 1 in magnitude as possible.
2. The pairing $x_i \rightarrow y_i$, must be avoided if the RGA element is negative i.e. when $\lambda_{ij} < 0$.

RGA analysis is performed for minimum phase and non-minimum phase system. The transfer matrix as discussed in (Johansson, May 2000)

$$G(s) = \begin{bmatrix} \dfrac{\gamma_1 c_1}{1+sT_1} & \dfrac{(1-y_2)c_1}{(1+sT_3)(1+sT_1)} \\ \dfrac{(1-y_1)2}{(1+sT_4)(1+sT_2)} & \dfrac{\gamma_2 c_2}{1+sT_2} \end{bmatrix}$$

Where $c_1 = \frac{T_1 k_1 k_c}{A_1}$ and $c_2 = \frac{T_2 k_2 k_c}{A_2}$, now by substituting the values for minimum and non-minimum from table 7-3 and 7-4:

Minimum phase system:

$$G(s) = \begin{bmatrix} \dfrac{2.6}{1+62s} & \dfrac{1.5}{(1+23s)(1+62s)} \\ \dfrac{1.4}{(1+30s)(1+90s)} & \dfrac{2.8}{1+90s} \end{bmatrix}$$

Non-minimum phase:

$$G(s) = \begin{bmatrix} \dfrac{1.5}{1+63s} & \dfrac{2.5}{(1+39s)(1+63s)} \\ \dfrac{2.5}{(1+56s)(1+91s)} & \dfrac{1.6}{1+91s} \end{bmatrix}$$

Refer appendix 13, where the MATLAB code of RGA is provided and figure 6-1, shows the output from the MATLAB code. Value closer to 1 should be selected for pairing of input and output, hence for minimum phase system (left side picture on figure 6-1) input u_1 with output y_1 and input u_2 with output y_2 can be paired. Similar, for non minimum phase system (right side picture on figure 6-1) input u_1 with output y_2 and input u_2 with output y_1 can be paired. Table 6-1 provides the final result of input output pairing in the form of table.

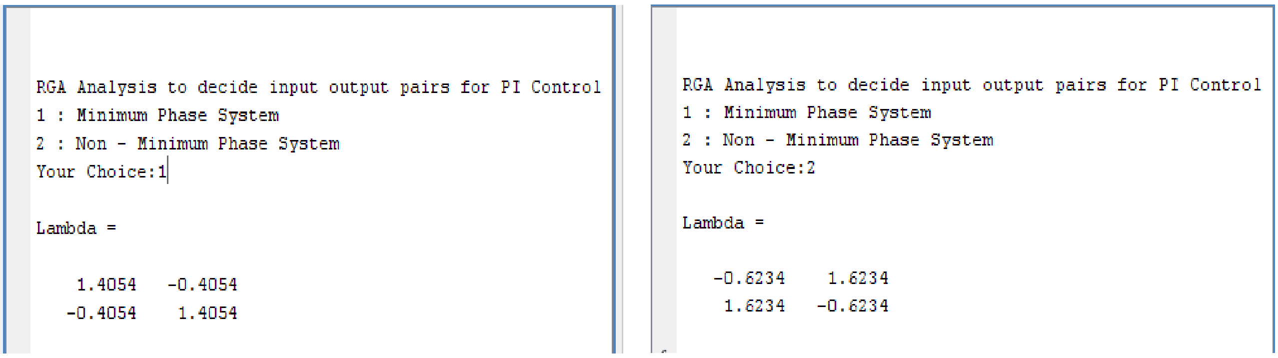


Figure 6-1: *RGA analyses for minimum and non-minimum phase of quadruple tank system.*

Table 6-1: *Pairing combination for minimum phase and non-minimum phase quadruple tank system*

Input	***Minimum phase***	***Non-minimum phase***
u_1 (Input 1, Pump 1)	y_1 (Output 1, Tank 1)	y_2(Output 2, Tank 2)
u_2 (Input 2, Pump 2)	y_2 (Output 2, Tank 2)	y_1(Output 1, Tank 1)

7. Problem Formulation

The main task of thesis is to investigate optimal control by simulation of non-linear process in MATLAB. Non-linear process doesn't satisfy the superposition principle or whose output is not directly proportional to its input. Most of the real life process or physical processes are non-linear process. Various non linear processes exist such as four tank level process, chemical reactors, distillation column etc. Four tank level process is chosen and optimal control is implemented and simulations are performed MATLAB.

This chapter is divided into various sections. Section 7.1 provides the introduction to four tank process, its inputs and outputs. Section 7.2 discusses the various steps in creating the state space model of the four tank process. Section 7.3 and section 7.4 provides the short introduction to minimum phase and non-minimum phase respectively. Section 7.5 deals with analysing the properties of a Linearized model of the system which is performed in MATLAB to find out the eigen values, observability and controllability of the system. If it satisfies the requirement then the Linearized model can be used for MPC and LQ optimal control simulation.

7.1 Four Tank Level Process

Four tank system is a Multiple input multiple output (MIMO) system. It's a non-linear system. The Linearized dynamics of the system have a multivariable zero that is possible to move along the real axis by changing a valve. The zero can be placed in both the left and right half plane, in this way the quadruple-tank process is ideal for illustrating many concepts in multivariable control (Johansson, May 2000).

The diagram of four tank process is shown in figure 7-1, there is a common reservoir from which water is pumped via pump p1 and p2. The voltages applied to pump are u1 and u2, these are the two input to the system. Table 7-1, provides the various inputs, outputs and state of the process which is common to minimum and non-minimum phase system.

Table 7-1: *Information on Inputs, outputs and states of Quadruple tank process*

Description	***Quantity***	***Description***	***Variables name (Refer figure 7-1)***
Number of Inputs	2	Voltages to Pump (Volts)	$u1\ and\ u2$
Number of Outputs	2	Water level in lower 2 tank (Centimetre), (The main objective is to control these two levels)	$y1\ and\ y2$
Number of states	4	Water level in all 4 tank (Centimetre)	$x1, x2, x3\ and\ x4$

The flow from pump 1 is divided by valve v1 to tank1 and tank 4, similarly flow from pump2 is divided by valve v2 to tank2 and tank4. This phenomenon is shown in table 7-2. All tanks interacts with each other, as all tanks have exit opening from which water is flowing out and this process is not measured, hence it can be observed as disturbance.

Table 7-2: *Flows to the tanks generated by the two pumps*

Tank / ***Pump***	***Tank1***	***Tank2***	***Tank3***	***Tank4***
Pump 1	$\gamma_1 k_1 v_1$	-	-	$(1-\gamma_1)k_1 v_1$
Pump2	-	$\gamma_2 k_2 v_2$	$(1-\gamma_{2)} k_2 v_2$	-

The main objective of the quadruple tank process is to control the level in lower tanks such as Tank 1 and Tank 2.

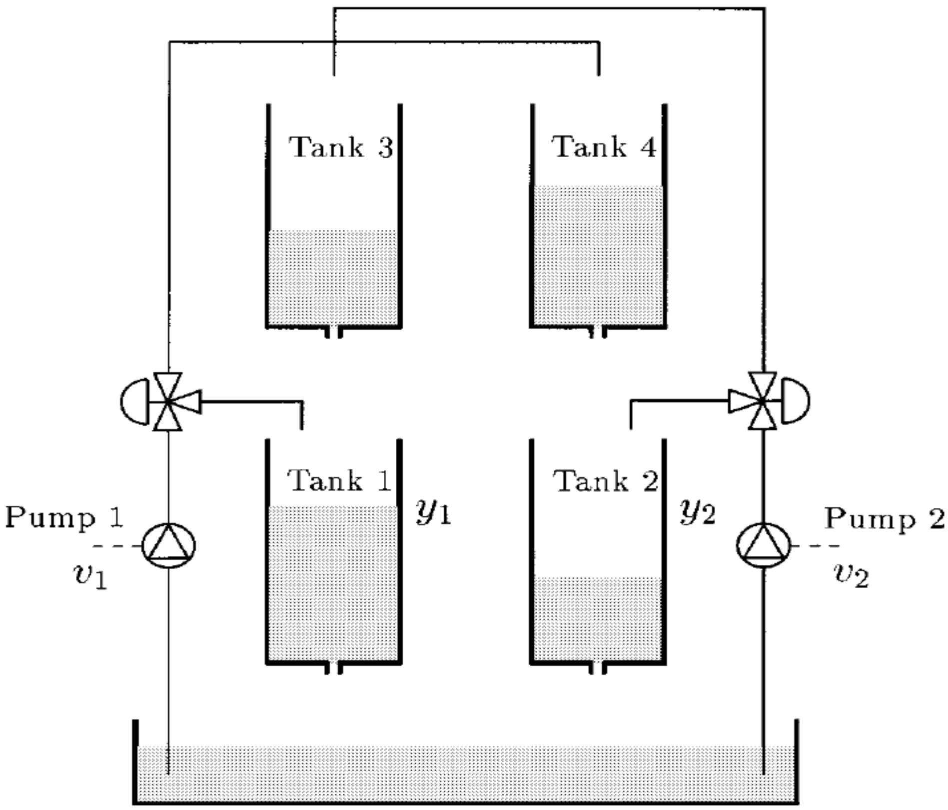


Figure 7-1: *Diagram of quadruple-tank process(Johansson, May 2000).*

7.2 Physical Model

Optimal control are based on model, hence a physical model is required to simulate the quadruple tank process. According to Nunes (2001) different type of model can be formed such as

- ***Impulse/Step response model***
  Models are obtained by simple experiment but it required a large amount of parameter.

- ***Transfer Function Model***
  Process model can be obtained on the basis of physical law or by system identification, less parameter needed than step response model.

- ***State Space Model***
  Generally used for linear time invariant system

For quadruple tank process continues state space Model is derived from the physical data.

As the process is non-linear, using mass balance and Bernoulli law the model is derived. The following equations are based on (Richter, 2010). Mass balance of tanks is given by:

$$\dot{V} \quad = q_{in} - q_{out} = a\,\dot{h} \tag{7.1}$$

With V= Volume water in Tank

a = Cross section area of the tank

h = water level in tank

q_{in} = inflow of water to tank

q_{out} = Outflow of water from tank

Bernoulli's Law which states

$$p + \frac{1}{2}\,\rho v_w^2 + \rho g h = const \tag{7.2}$$

At water surface ($v_w = 0$) and at the bottom of each tank ($h = 0$) and subtracting the resulting equations from each other results in

$$q_{out} = a_o v_w = a_o \sqrt{2g}\sqrt{h} \tag{7.3}$$

Where

a_o = cross section area of an outlet

v_w = speed of water outflow

H = water level in tank

G = acceleration due to gravity

Pump generated flow is proportional to the applied voltage, hence $q_{pump} = K_p . v_1$, where K_p is pump gain.

The following ideas , equation and physical data are based on (Johansson, May 2000). Non-linear Mathematical model in the form of differential equations are derived from the physical data of the quadruple tank process.

$$\frac{dh_1}{dt} = -\frac{a1}{A1}\sqrt{2gh_1} + \frac{a3}{A1}\sqrt{2gh_3} + \frac{\gamma_{1k_1}}{A_1}\,v_1 \tag{7.4}$$

$$\frac{dh_2}{dt} = -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_{2k_2}}{A_2}\,v_2 \tag{7.5}$$

$$\frac{dh_3}{dt} = -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}\,v_2 \tag{7.6}$$

$$\frac{dh_4}{dt} = -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_4}\, v_1 \tag{7.7}$$

The parameter $\gamma_1, \gamma_2 \in (0,1)$ are determined from how valves are set prior to an experiment. The common parameter values of the process are given in table 7-3.

Table 7-3: *Common Parameter for Quadruple Tank process(Johansson, May 2000)*

Parameter	***Unit***	***Description***	***Value***
A_1	cm^2	Cross Sectional Area of Tank 1	28
A_1	cm^2	Cross Sectional Area of Tank 2	32
A_1	cm^2	Cross Sectional Area of Tank 3	28
A_1	cm^2	Cross Sectional Area of Tank 4	32
a_1	cm^2	Cross Sectional Area of the outlet hole of Tank 1	0.071
a_2	cm^2	Cross Sectional Area of the outlet hole of Tank 2	0.057
a_3	cm^2	Cross Sectional Area of the outlet hole of Tank 3	0.071
a_4	cm^2	Cross Sectional Area of the outlet hole of Tank 4	0.057
K_c	V/cm	Gain of Level measurement for tank 1 and tank 2	0.50
g	cm/s^2	Gravity	981

The quadruple tank can be studied based on two operating point such as minimum and non-minimum phase system (discussed in detail in section 7.3 and 7.4), the chosen operating values of certain parameter are shown in table 7-4.

Table 7-4: *Operating condition of quadruple tank process*

Parameter	***Unit***	***Description***	***Minimum Phase***	***Non-minimum phase***
h_1	cm	Initial height of tank 1	12.4	12.6
h_2	cm	Initial height of tank 2	12.7	13.0
h_3	cm	Initial height of tank 3	1.8	4.8
h_4	cm	Initial height of tank 4	1.4	4.9
v_1	volt	Initial voltage to pump 1	3	3.15

v_2	volt	Initial voltage to pump 2	3	3.15
k_1	cm^3/Vs	Gain of Pump 1	3.33	3.14
k_2	cm^3/Vs	Gain of Pump 2	3.35	3.29
γ_1	-	Setting of Valve 1 for tank 1 and 4	0.70	0.43
γ_2	-	Setting of Valve 2 for tank 2 and 3	0.60	0.34

As non linear system can be converted to Linearized state space equation by:

$$A = \begin{bmatrix} \frac{-1}{T_1} & 0 & \frac{A_3}{A_1 T_3} & 0 \\ 0 & \frac{-1}{T_2} & 0 & \frac{A_4}{A_2 T_4} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{T_3} & 0 \end{bmatrix} \quad B = \begin{bmatrix} \frac{\gamma_1 k_1}{A_1} & 0 \\ 0 & \frac{\gamma_2 k_2}{A_2} \\ 0 & \frac{(1-\gamma_2) k_2}{A_3} \\ \frac{(1-\gamma_1) k_1}{A_1} & 0 \end{bmatrix} \quad D = \begin{bmatrix} K_c & 0 & 0 & 0 \\ 0 & K_c & 0 & 0 \end{bmatrix}$$

Where time constantT_i, for i=1, 2,3 and 4 refers to $T_i = \frac{A_i}{a_i}\sqrt{\frac{2\, h_i}{g}}$.

The above equation can be formulated to state space equation and 'x' refers to the state variable which is the height of tanks.

$$\dot{x} = Ax + Bu$$

$$y = Dx$$

7.3 Minimum Phase

A system with all poles and zeros inside the unit circle, both the system function and inverse is casual and stable (Arslan, 2005). Whose transfer function have all pole and zero in the left half of S-plane (D.Viswanath, 2011)

According to Johansson (2000), the condition for minimum phase is $1 < \gamma_1 + \gamma_2 < 2$, where $\gamma_1 and\ \gamma_2$determined from how valves are set prior to an experiment. Let q_i denote the flow through pump i and assume $q_1 = q_2$. Then the sum of the flows to the upper tank is $[2 - (\gamma_1 + \gamma_{2)}]q_1$ and the sum of the flow to the lower tank is $(\gamma_1 + \gamma_2)q_1$. Hence, the flow to the lower tank is greater than the flow to the upper tanks if $\gamma_1 + \gamma_2 > 1$, such that if the system is minimum phase. Refer table 7-4 for the various values of the minimum phase system.

7.4 Non-Minimum Phase

Non-minimum phase systems are those whose transfer function have one or more poles or zero in the right half of the S-plane (D.Viswanath, 2011).

According to Johansson (2000), the condition for minimum phase is $0 < \gamma_1 + \gamma_2 < 1$. The flow to the lower tank is smaller than the flow to the upper tanks if the system is in non-minimum phase. Refer table 7-4 for the various values of the non minimum phase system.

Control Engineers have been aware of non-minimum phase systems showing either undershoot or time delay characteristics for some considerable time (Urgen and Schoor, 2011)

7.5 Properties of Linearized Model

The given non-linear model was converted to a linear model as described in section 7.2. Refer figure 7-2 for the various values of the simulation results of the MATLAB programming as provided in appendix 2. The main criteria to check a linear model are

- ***Eigen Value:*** If the eigenvalues are all real, distinct and have negative values (which means they lie in the negative half of the complex plane) then the system is stable (Ruscio, 1996). With reference to figure 7-2 the eigenvalues for minimum and non-minimum phase are distinct and negative, hence the system is stable. Both systems have four time constant.

- ***Controllability:*** The system is controllable if the controllability matrix has full rank (Ruscio, 1996). The controllability matrix was found and then rank was found for minimum and non-minimum phase system. The rank is 4, which is full rank which means all state variables are controllable.

- ***Observability:*** The system is observable if the observability matrix has full rank (Ruscio, 1996). The observability matrix was found and then rank was found for minimum and non-minimum phase system. The rank is 4, which is full rank.

- ***Transmission zero:*** The position of the valves determine the location of a multivariable zero of the Linearized model, the zero can be put in either left or the right half plane (Johansson, May 2000). For minimum phase system the zeros are in the left half of the complex plane whereas for non-minimum phase system the one zero is in right half and the other in left half of the plane. According to Johansson (May 2000) this quadruple tank process which is ideal to show performance limitation due to zero in right half plane.

Refer figure 7-3, for the various steps or various simulation performed till now in this thesis.

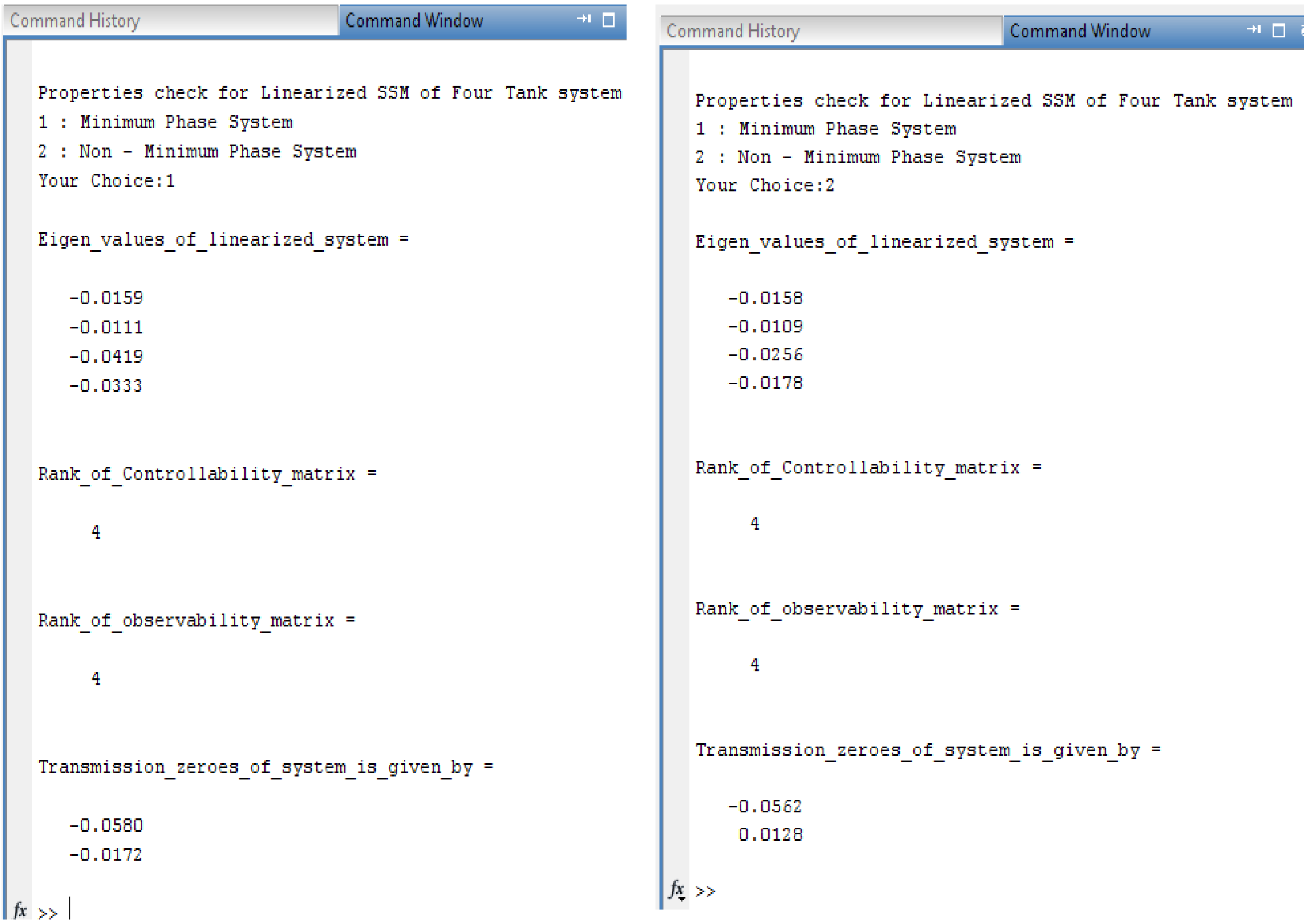


Figure 7-2: *Linear model analysis of quadruple tank system, left shows the minimum phase system and right show non-minimum phase properties.*

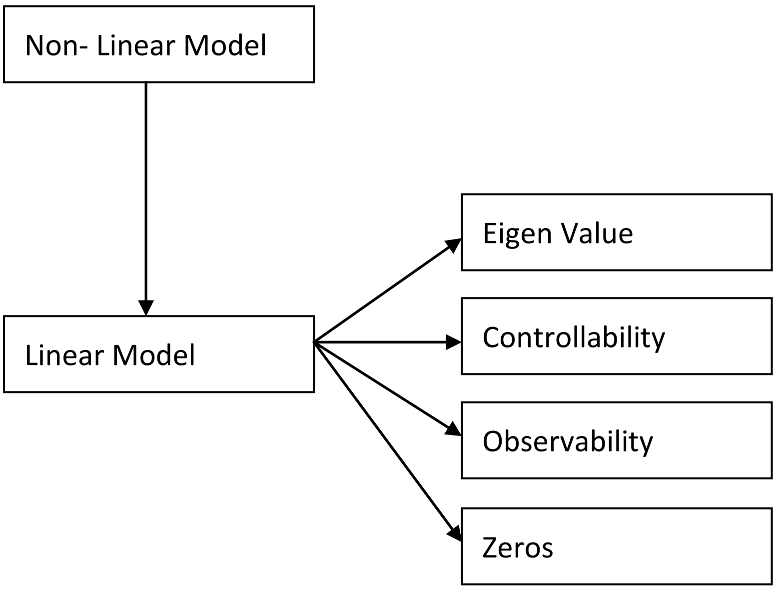


Figure 7-3: *Various steps performed in this thesis so far.*

8. Problem Solution (Matlab Simulation)

This chapter contains the simulation results to the problem mentioned in chapter 7 which is a four tank system. Mainly LQ optimal control and MPC control with integral actions are simulated. Refer figure 8-1, which represents the various type of MATLAB programming implemented for the four tank system. LQ optimal control is implemented using unconstrained and constraints (if-else loop), whereas MPC with integral action is implemented using an algorithm based constraints, if else loop constraints and unconstrained. PID (using RGA analysis) controller simulation is also performed as an additional task. By using Kalman filter in simulation unmeasured states are predicted (tank level for tank 3 and tank 4).

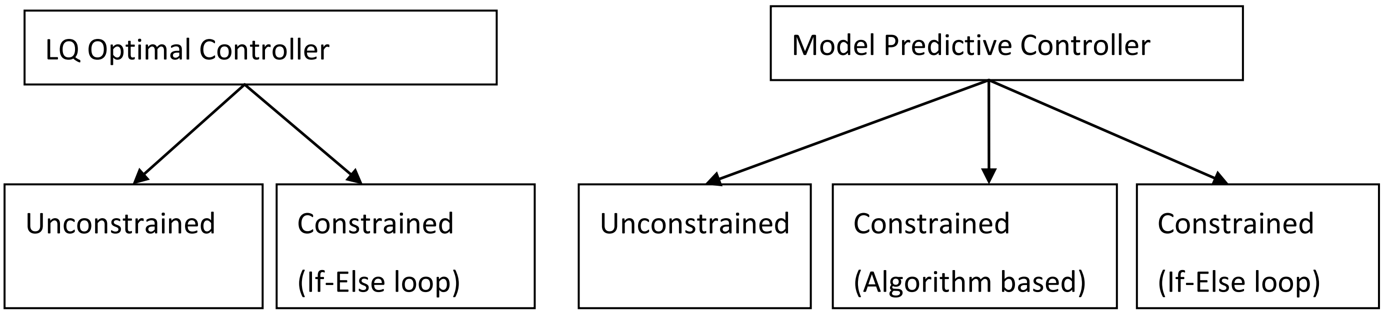


Figure 8-1: *Various type of simulation performed in MATLAB for quadruple tank process using minimum and non-minimum phase system.*

This chapter is divided into three section, where section 8.1 provides the results of LQ optimal control which is again divided into subsection as 8.1.1 for minimum phase system and 8.1.2 non-minimum phase system, in both these subsection the comparison is based on constraints and unconstrained LQ optimal control. Section 8.2 provides the results of MPC which is again divided into subsection as 8.2.1 for minimum phase system and 8.2.2 non-minimum phase system, in both these subsection the comparison is based on constraints and unconstrained MPC. Section 8.3 provides the comparison based on constraints for controllers like LQ, MPC and PI for minimum phase system.

8.1 LQ Optimal Control (Constraints handling Comparison)

Theoretical description of LQ optimal control with integral action is discussed in section 2.3 and constraint handling are discussed in section 4.4. Based on these theoretical description and equation LQ optimal control is implemented on the four tank process. The simulation of LQ optimal control can be divided into 4 types depending on constraints, minimum or non-minimum phase system of four tank process refer figure 8-2. Use of Kalman filter is for state estimation and reduction of noise within the model.

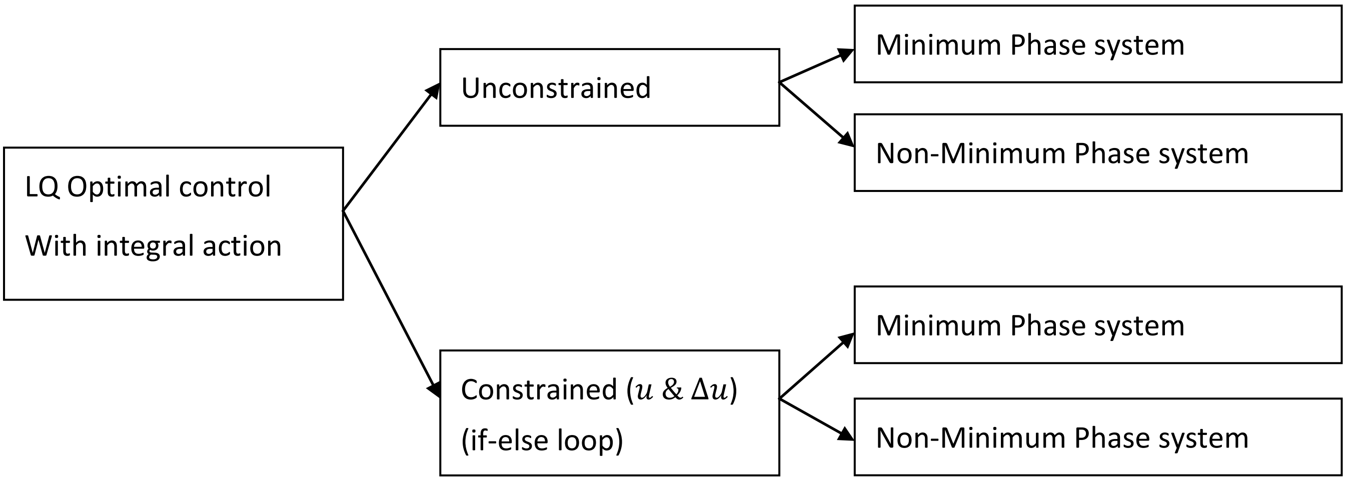


Figure 8-2: *Different types of LQ optimal control simulation implemented on four tank process.*

Tuning of LQ optimal control depends on selection of weighting matrices $R_w \; and \; Q$, which is based on trial and error selection method. The main calculation of optimal control is based on a sub program named "dlqdu_pi.m" created by Ruscio (2012a). And inside this program "dlqr" is used to find the feedback/gain matrix.

The main task of the thesis is to deal with constraint handling, hence simulation result for specific programming like minimum phase system with constraint or non-minimum phase system with unconstrained are rather not provided, whereas only the comparison plot of tank 1 level of minimum phase system with constrained and unconstrained are compared similarly for tank 2 and for control signal etc. As discussed in section 4.4, constraints are implemented in LQ optimal control with if-else loop only.

The flow chart of the LQ optimal control programming using constraint and unconstrained method is provided in figure 8-3 and figure 8-4.

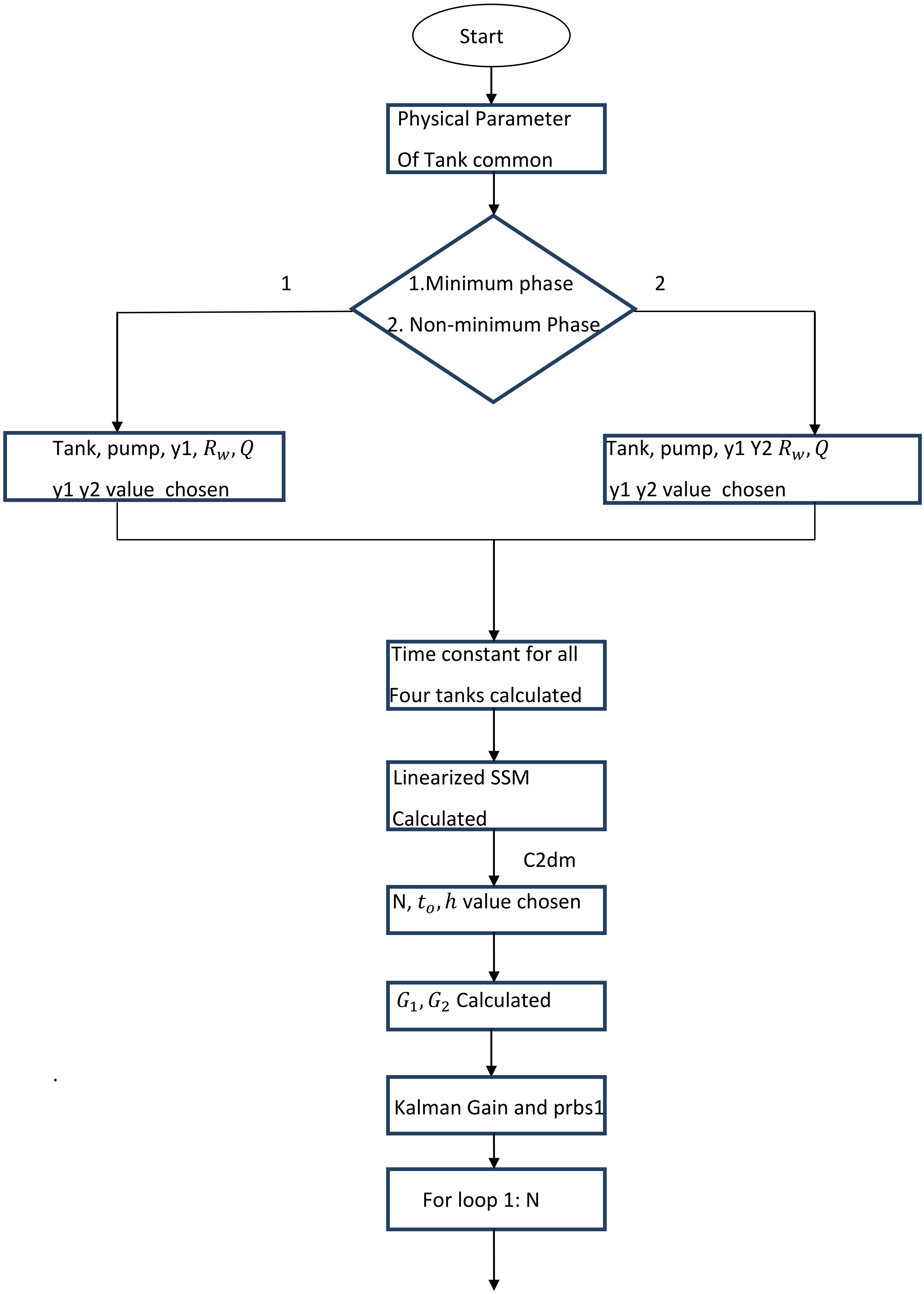


Figure 8-3: *Flow chart of LQ optimal control programming Part I.*

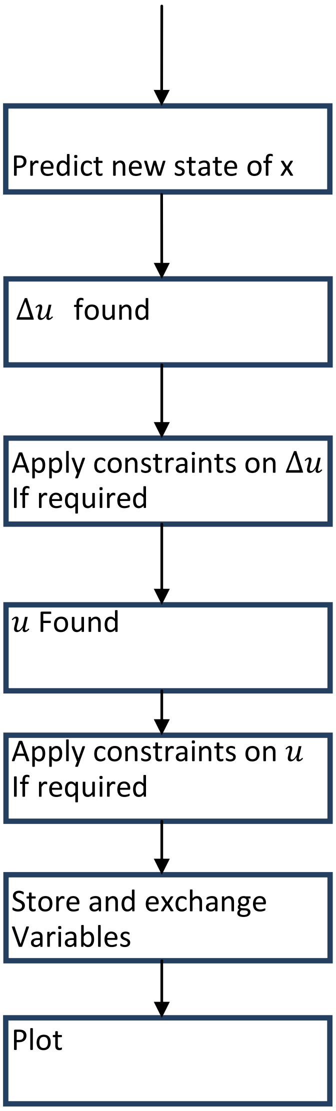


Figure 8-4: *Flow chart of LQ optimal control programming part II.*

As shown in the above flowchart, the main difference between Constrained and Unconstrained LQ optimal control is hard limit constraints for U and ΔU , which is implemented in MATLAB coding via if-else loop for Tank 1 and Tank 2.

8.1.1 Minimum Phase system Comparison

This section provides the comparison of LQ optimal control with integral action on a minimum phase quadruple tank system based on constraint and unconstrained. Two separate MATLAB programs are simulated for LQ optimal control with unconstrained (refer appendix 3) and for LQ optimal control with constrained (refer appendix 4).

Instead of showing two sets of plotting based on constrained and unconstrained, single plotting is shown which combines both the data's of constraints and unconstrained. The above two MATLAB coding was simulated separately and the data was transferred to excel file (this concept applies to section 8.1.2 also). Refer appendix 5 for the MATLAB code, this code helps to write the data's to excel file, reading from excel file and plotting it. The main parameters for the simulation of this task are provided in table 8-1.

Table 8-1: *Critical parameter for simulation for LQ optimal control for minimum phase system*

N	R_w	***Q***	***Constraints on Control signal U (Pump voltage) [Volts]***				***Tank 1 Set point***	***Tank 2 Set point***
			U_{max}	U_{min}	ΔU_{max}	ΔU_{min}		
1500	10*[1,0;0,1]	0.1*[100,0;0,1]	5	0	0.4	-0.4	12.5 and 12.3 cm	12.8 and 12.6 cm

Above parameters were used for simulation and refer figure 8-5, which shows the tank 1 and tank 2 level with respect to reference for constrained and unconstrained LQ optimal control. X axis denotes the time stamp and y axis denote the level in centimetre. One important point observed in this plot was the output of both tank level were following the reference/set point level, but there is no change in output level for tank 1 and tank 2 for constrained and unconstrained LQ because the change in ΔU or U in constrained LQ optimal was within the constraints limits refer figure 8-6 and 8-7. The major difference between the unconstrained and constrained LQ is the if else loop, which is used to implement the constraints, as the ΔU and U values are within the constraint limit, there is no difference observed in the performance of constrained and unconstrained LQ optimal control. In the plot, only red line is shown whereas red (constrained) and green (unconstrained) are overlapped with each other. Tank 1 and tank 2 are stable and controllable.

Figure 8-6 shows the control signal U plot with respect to time stamp. X-axis denotes the time and y-axis denotes the voltage to pump. In this plot, it is observed that LQ optimal control for constrained and unconstrained remains same as they are overlapped.

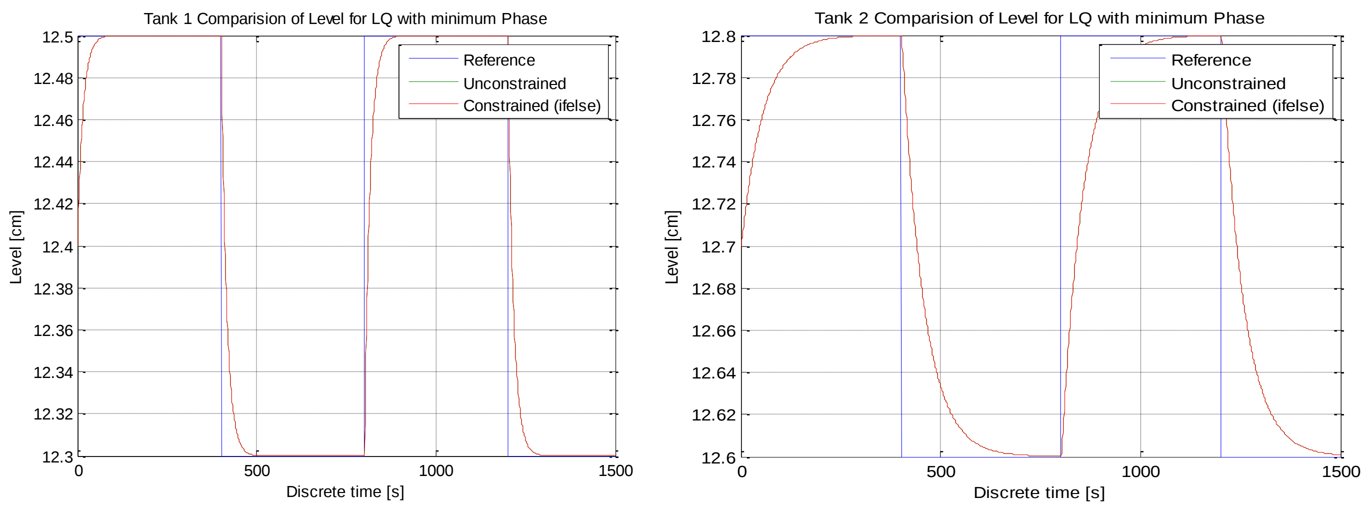


Figure 8-5: *Level of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.*

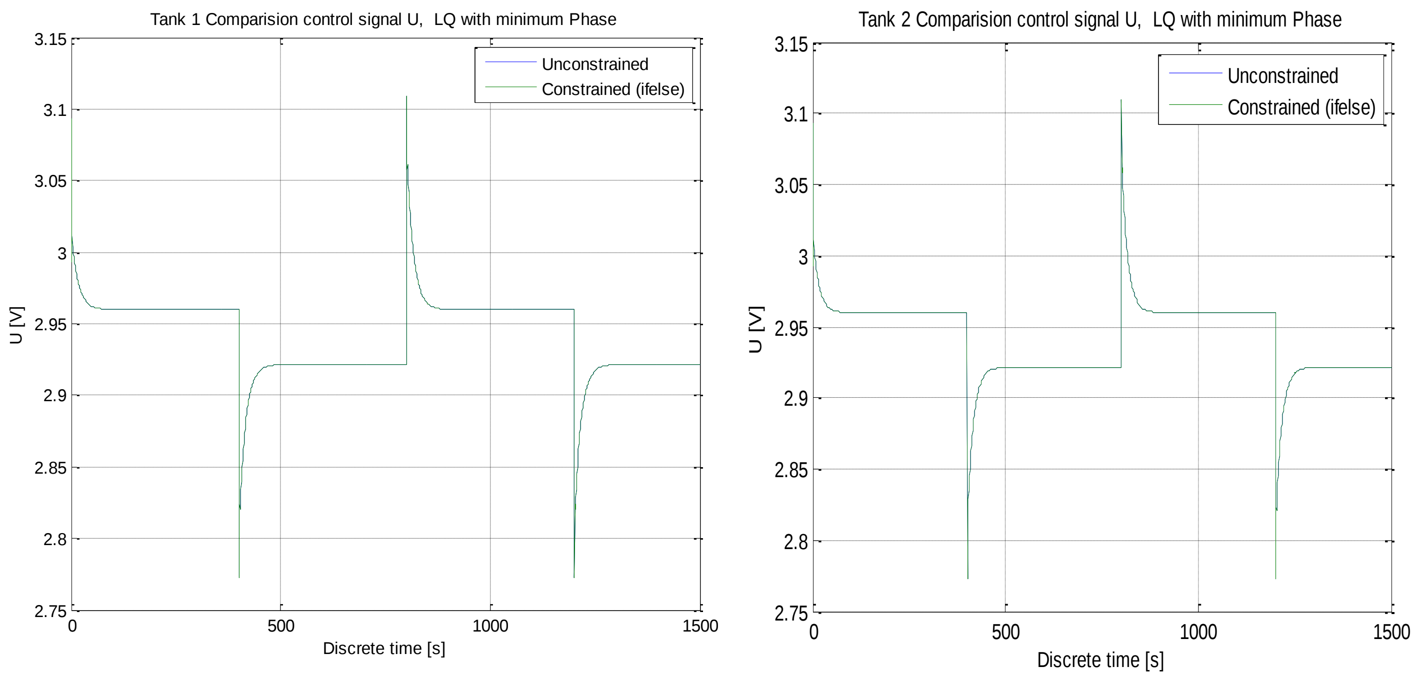


Figure 8-6: *Control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.*

Figure 8-7 which shows the comparison of LQ unconstrained and constrained plot of, change in control signal ΔU for tank 1 and tank 2. In this plot, it is observed that there is no change or deviation in change in control signal for constrained and unconstrained LQ as they both overlap with each other.

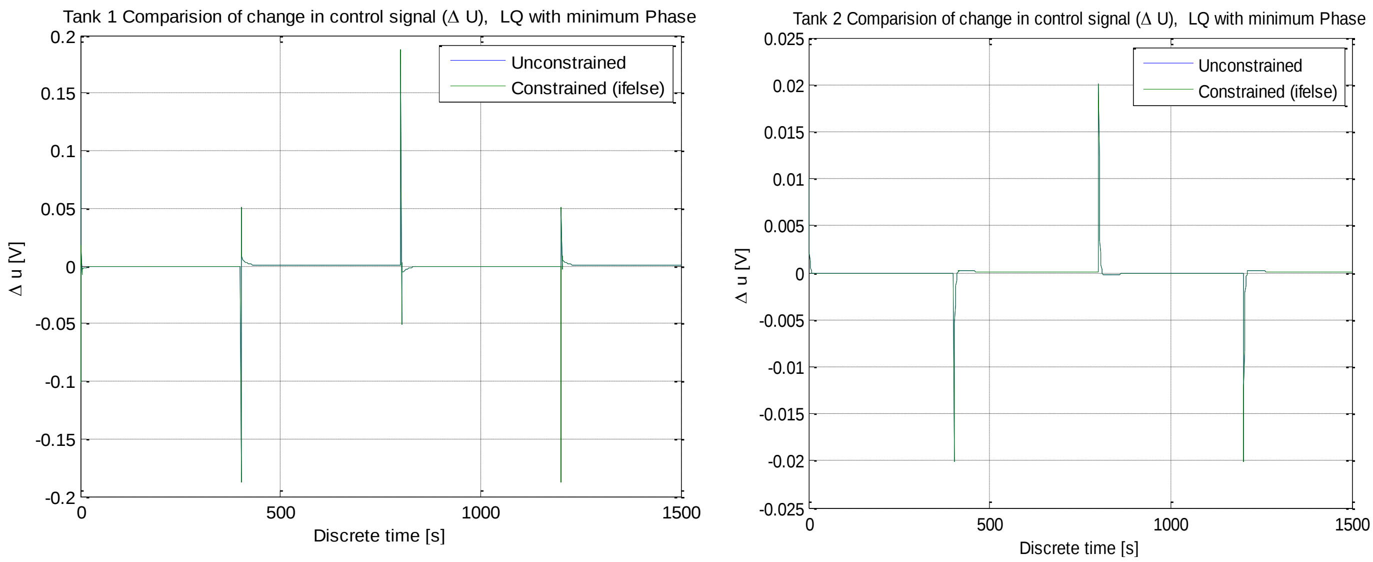


Figure 8-7: *Change in control signal ΔU of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller*

Remark 1: *Based on the above comparison for constrained and unconstrained LQ optimal control for minimum phase system of quadruple tank system, it is observed that constrained (if else loop) and unconstrained LQ optimal controller provide the same result because when the constraints are implemented in the LQ optimal control, the value of* U *and* ΔU *were within the constraint limit, hence no major change in output level control of tank 1 and tank 2 are observed for constrained and unconstrained LQ optimal controller. Tank 1 and Tank 2 level are stable and controllable for minimum phase quadruple tank process. The changes in* Δu *are very small. Control signal U, change in control signal* ΔU *and the output level for tank 1 and tank 2 overlap with each other for constrained and unconstrained LQ optimal control.*

8.1.2 Non-Minimum Phase system Comparison

This section provides the comparison of LQ optimal control with integral action on a non-minimum phase quadruple tank system based on constraint and unconstrained. Two separate MATLAB programs are simulated for LQ optimal control with unconstrained (refer appendix 3) and for LQ optimal control with constrained (refer appendix 4). The MATLAB codes for plotting the results are provided in appendix 6. The main parameters for the simulation of this task are provided in table 8-2.

Table 8-2: *Critical parameter for simulation for LQ optimal control for non minimum phase*

N	R_w	***Q***	***Constraints on Control signal U (Pump voltage) [Volts]***				***Tank 1 Set point***	***Tank 2 Set point***
			U_{max}	U_{min}	ΔU_{max}	ΔU_{min}		
1500	[.1,0;0,.1]	[.01,0;0,.001]	5	0	0.4	-0.4	12.5 and 12.3 cm	12.8 and 12.6 cm

As four tank process are very interactive process, change in one set point in one tank affects the level in another tank. The tuning parameter R_w and Q had been modified to match the non-minimum phase process because when the value of minimum phase was used, the system was more unstable and uncontrollable, hence these matrices are modified. Constraints limit, set point level and number of sample remain same. Refer figure 8-8, it is observed that tank 1 is closely following the reference, whereas Tank 2 is controllable to a slight extent, as the deviation observed is 0.15 cm of water level. In this case also LQ constrained and LQ unconstrained values are overlapped, hence in the plot only one curve is visible. The reason for deviation in level in tank 2 is due to the non minimum phase whose zero is in the right half of the plane and control engineers have been aware of non-minimum phase systems showing either undershoot or time delay characteristics for some considerable time (Urgen and Schoor, 2011).

Figure 8-9 the change of control signal U values are plotted and it is observed that the values are also overlapped here for constrained and unconstrained LQ, similarly in figure 8-10 where changes in control signal ΔU are also overlapped. Reason for overlapping of values is mainly due to constrained LQ where the values of U and ΔU where within the constraint limits.

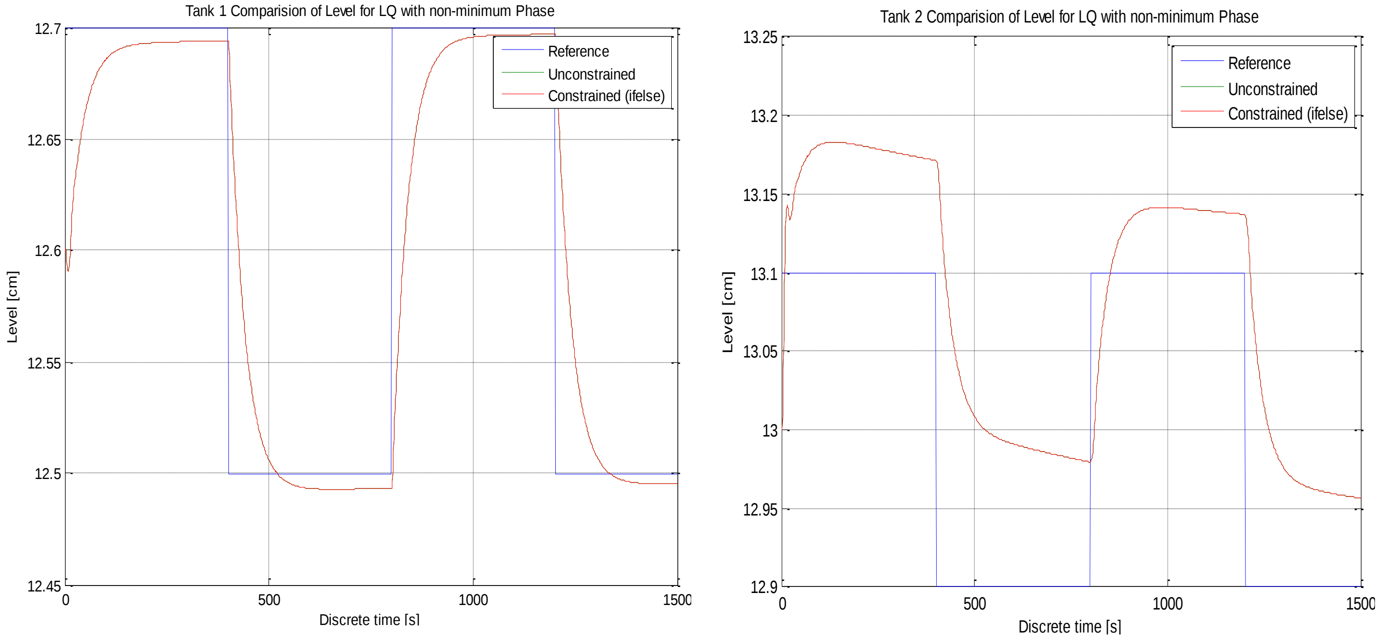


Figure 8-8: *Level of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller*

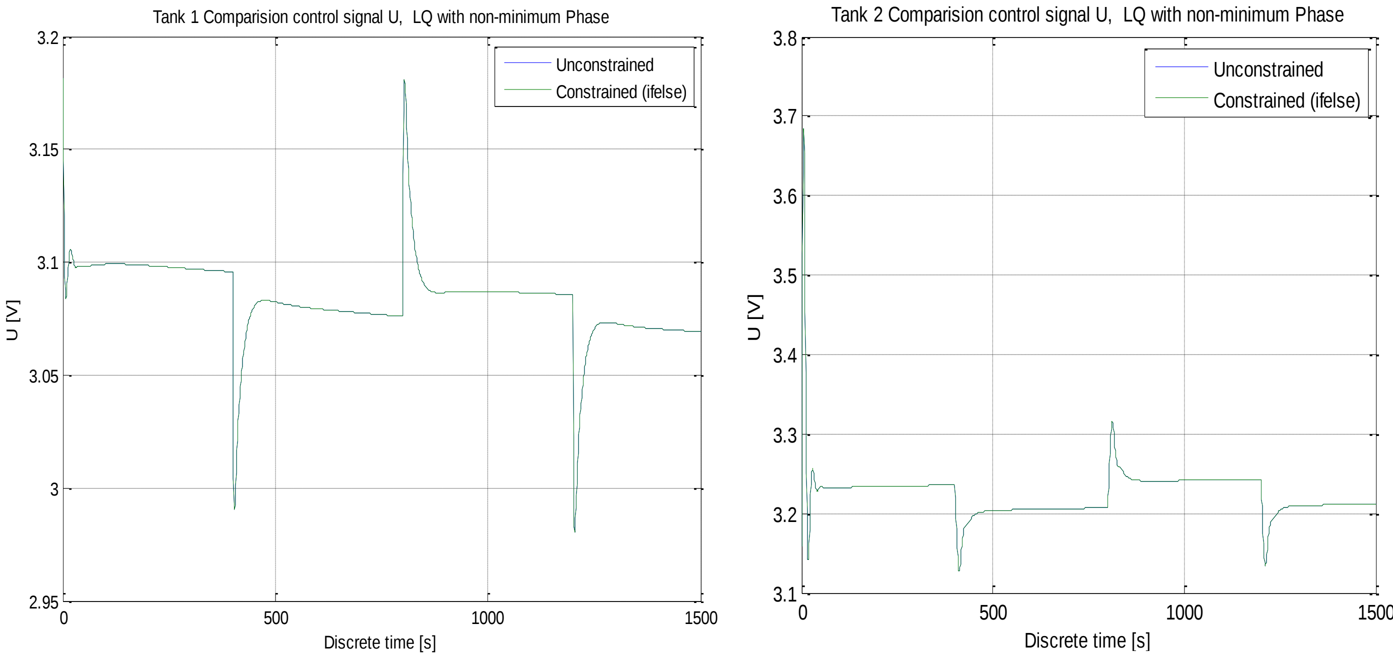


Figure 8-9: *Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.*

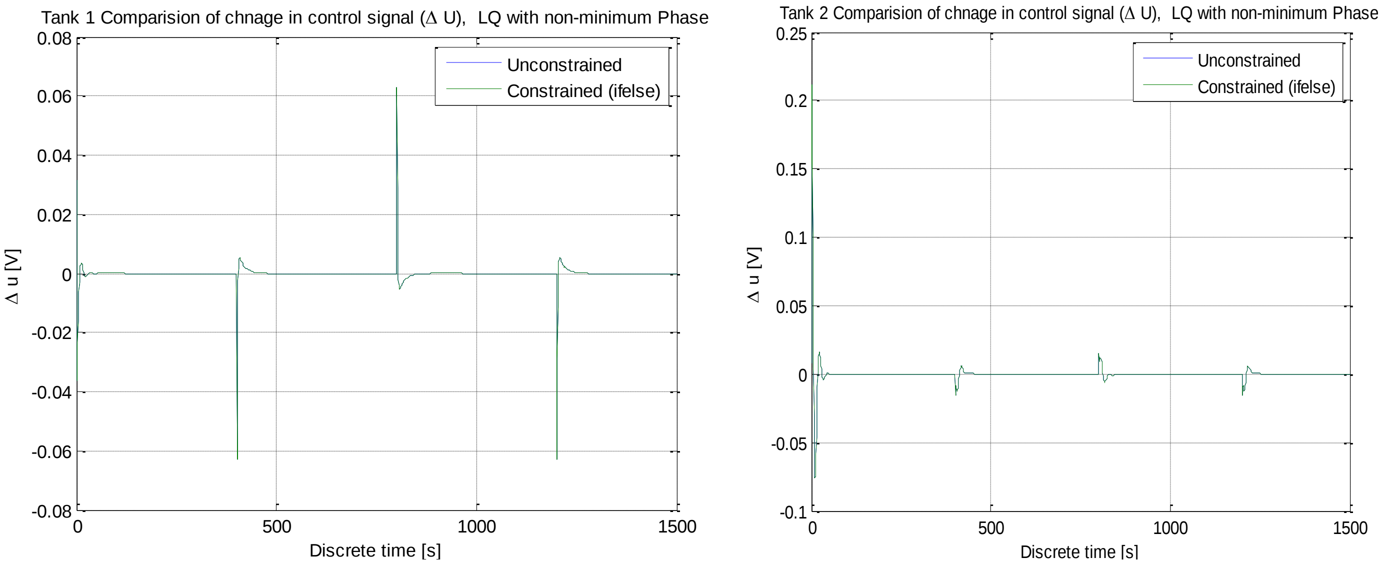


Figure 8-10: *Change in control signal ΔU of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained LQ optimal controller.*

Remark 2: *Based on the above comparison for constrained and unconstrained LQ optimal control for non minimum phase system of quadruple tank system, its observed that constrained (if else loop) and unconstrained LQ optimal controller provide the same result because when the constraints are implemented in the constrained LQ, the value of U and ΔU were within the constraint limit, hence no major change in output level control of tank 1 and tank 2 observed for constrained and unconstrained LQ optimal controller. Tank 1 level was controllable whereas tank 1 level observed a deviation of 0.15 cm level difference. The changes in Δu is very small, hence changes in U were also small.*

No difference observed in LQ constrained and unconstrained algorithms for minimum and non-minimum phase quadruple tank system.

8.2 MPC (Constraints Handling Comparison)

Theoretical description of MPC control with integral action is discussed in section 3.2 and constraint handling is discussed in section 4.3. Based on these theoretical description and equation MPC is implemented on the four tank process. The simulation of MPC with integral action can be divided into 6 types depending on constraints and minimum or non-minimum phase systems of four tank process refer figure 8-11. Use of Kalman filter is for state estimation and reduction of noise within the model.

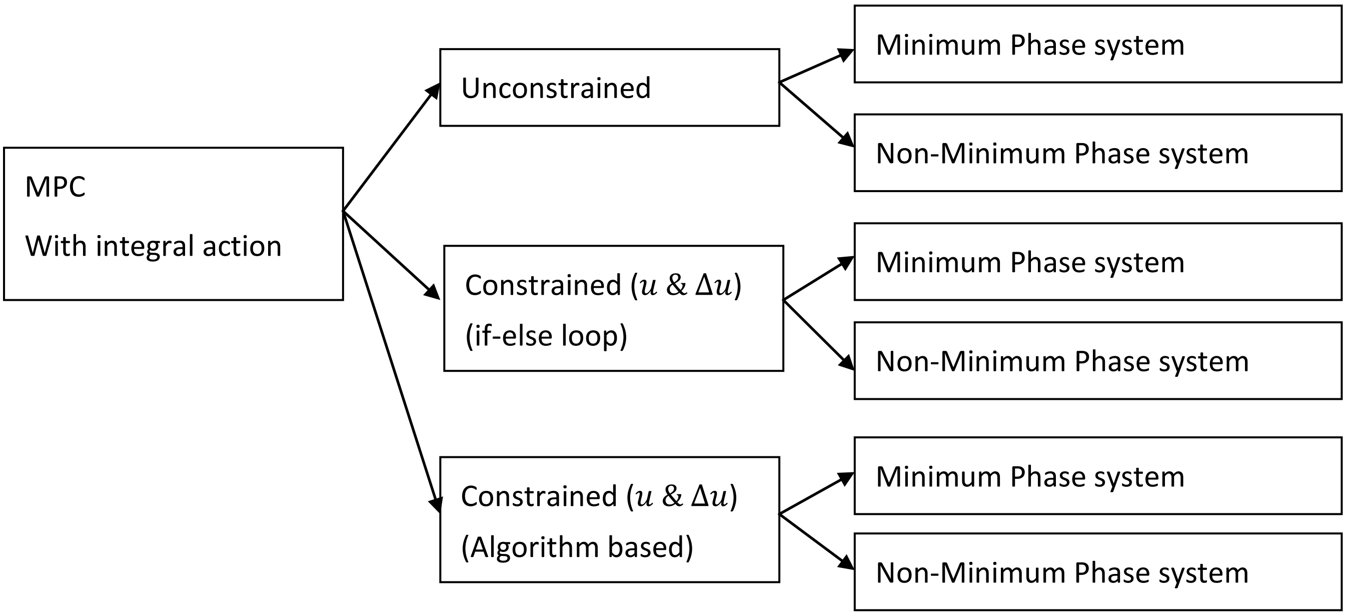


Figure 8-11: *Different types of LQ optimal control simulation implemented on four tank process.*

Tuning of MPC control depends on selection of weighting matrices R, Q and prediction Horizon L. Selections of matrices $R \text{ and } Q$ are based on trial and error methods. Constraints also play an important role in MPC, constraints are chosen based on the process requirements. Depending upon the programming type different MATLAB codes are generated. Refer appendix 7 for MATLAB code for MPC Constrained (Algorithm based) with integral action, appendix 8 for MATLAB code for MPC Constrained (if else loop) with integral action and appendix 9 for MATLAB code for MPC unconstrained with integral action.

As discussed earlier, a comparison plot where tank 1 level of minimum phase system with constrained and unconstrained simulations are compared, similarly for tank 2 also. Control signal U and change in control signal ΔU are also compared. The flow chart of the MPC programming using constraint and unconstrained method is provided in figure 8-12 and figure 8-13.

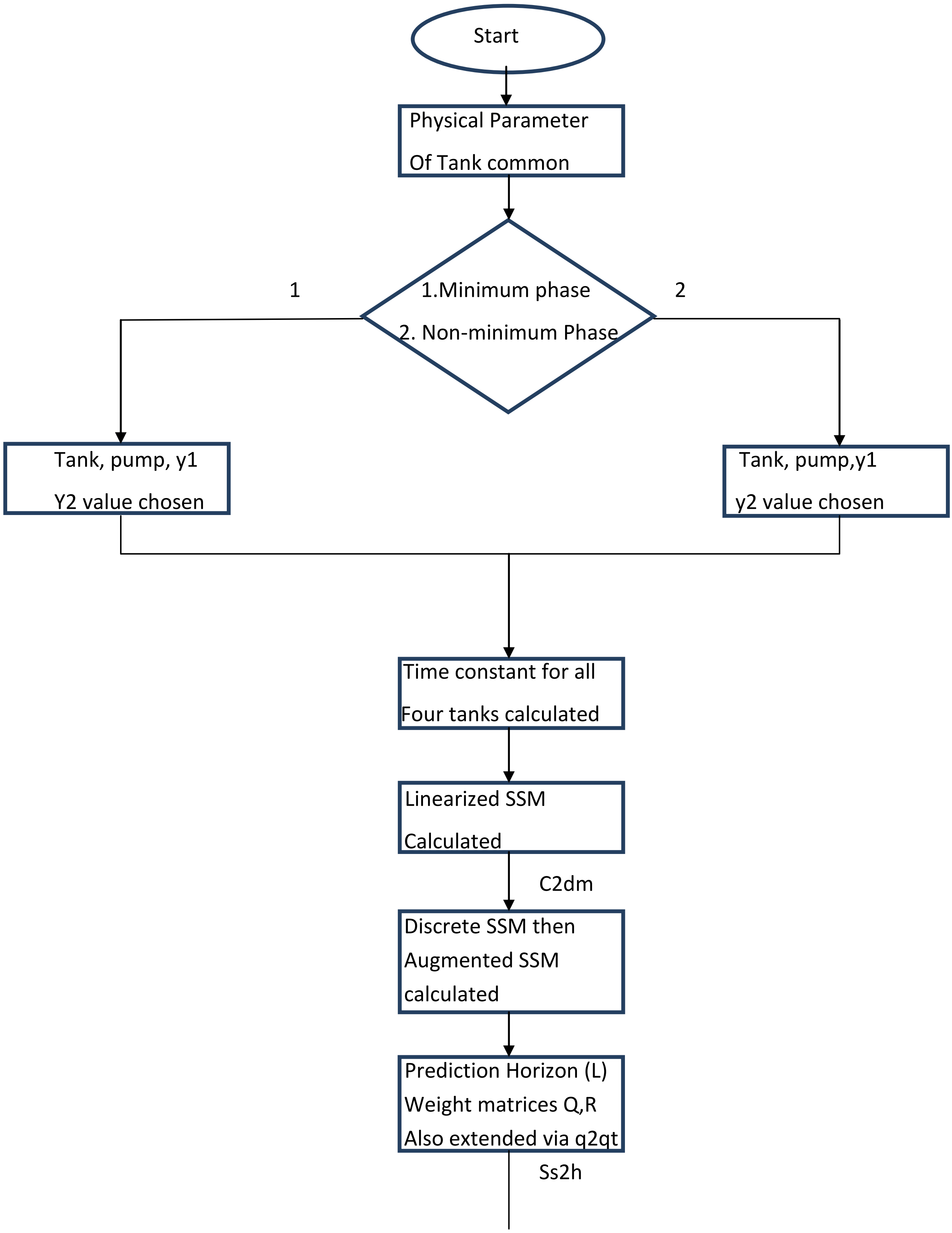


Figure 8-12: *Flow chart of MPC with integral action programming part I.*

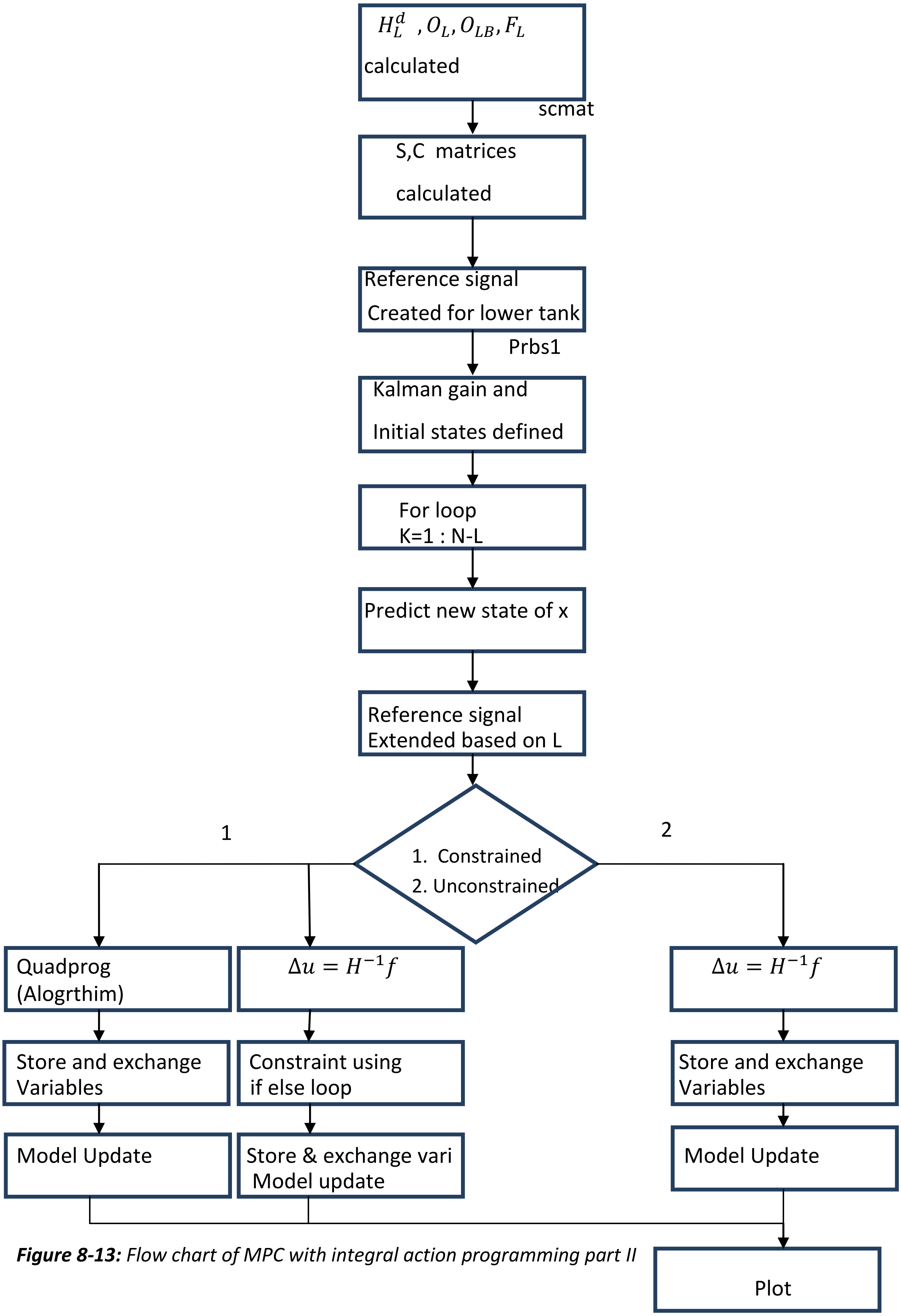


Figure 8-13: *Flow chart of MPC with integral action programming part II*

As shown is flowchart, MPC provides a possibility to handle constraints in the algorithm itself and also through hard limit constraints (if else loop). The unconstrained MPC can be compared to LQ optimal control.

8.2.1 Minimum Phase

This section provides the comparison of various MPC techniques on minimum phase quadruple tank process. Three different MATLAB code was simulated namely Unconstrained MPC, constrained MPC (Algorithm based) and Constrained MPC (if else loop), these data's were written to a excel file and plotted, coding provided on appendix 10.

Table 8-3, provides the list of critical parameter which were used in simulation of MPC controller, these values were common to different types of simulation. The matrices R and Q were selected on trial and error method.

Table 8-3: *Critical parameter for MPC simulation on minimum phase quadruple tank process.*

N	***L***	***R***	***Q***	***Constraints on Control signal U (Pump voltage) [Volts]***				***Tank 1 Set point***	***Tank 2 Set point***
				U_{max}	U_{min}	ΔU_{max}	ΔU_{min}		
250	8	0.09*eye(2)	50*eye(2)	5	0	0.4	-0.4	14.1 & 13.9 cm	12.1 and 11.9 cm

Figure 8-14 provides a comparison plot of difference type of MPC simulation, where Tank 1 and tank 2 level are compared with reference level. Different colour coding clearly distinguishes the type of controllers. As all outputs change their state before the reference signal changes, hence all simulation follows MPC main behaviour. With reference to figure 8-14, it is clearly visible that unconstrained MPC follows the reference much faster than other types of controller and also it has less overshoots compared to others.

Hence from the figure 8-14, its understood that Unconstrained MPC settles/follows the reference much faster, then its constrained (Algorithm) based MPC which follows the reference and with little delay and little overshoot compared to other controllers, finally constrained (if else loop) MPC follows the reference with big undershoot.

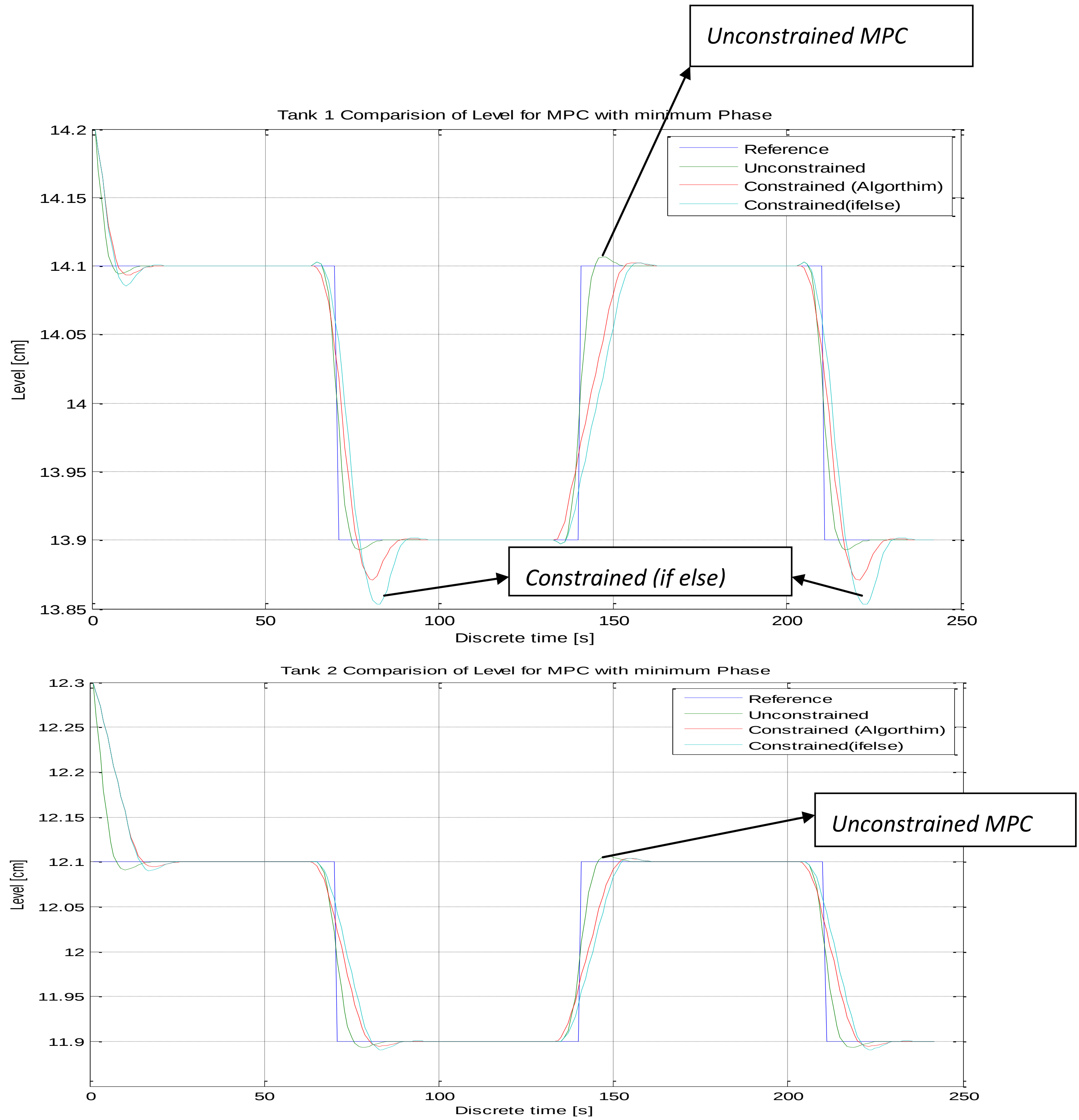


Figure 8-14: *Level of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action*

Referring to plot 8-15, which shows the plotting of input control signal U, which is the voltage to pump is compared with different type of MPC. The constrained applied on control signal U is $0 \leq U \leq 5$, this is due to physical/mechanical constraint on instruments. This constraint is valid for both tanks. But figure 8-15 shows that the unconstrained MPC which provides a control signal U maximum of nearly 8.1 volt and minimum of nearly -0.08 V, which violates the required constraints limit on control signal.

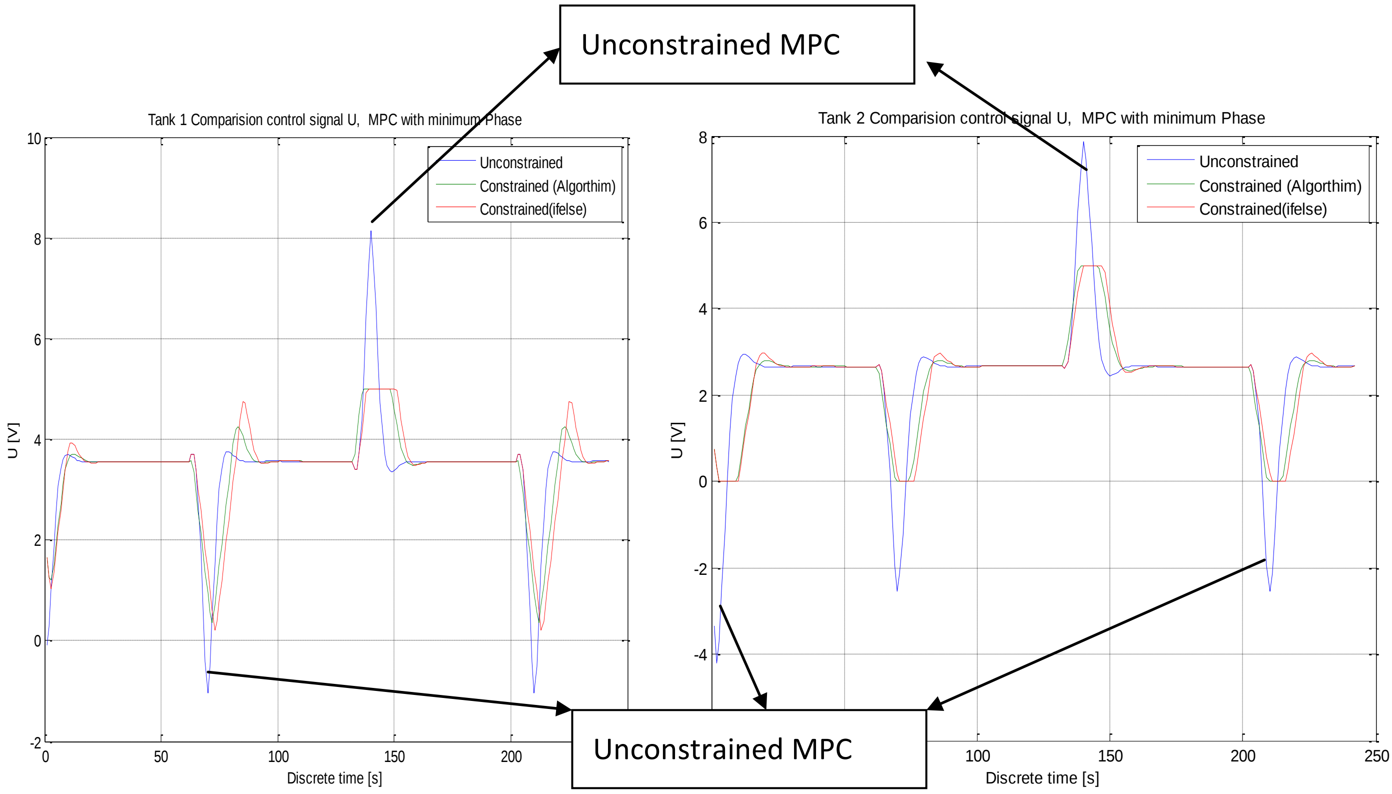


Figure 8-15: *Control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action*

In the same figure 8-15, it clearly shows that constrained MPC (algorithm and if else loop) follows the applied constraints, whenever unconstrained violated the constrained limit and rise up to 8v, the control signal was 5v for constrained MPC. This clearly shows constrained MPC are following the constraints limit.

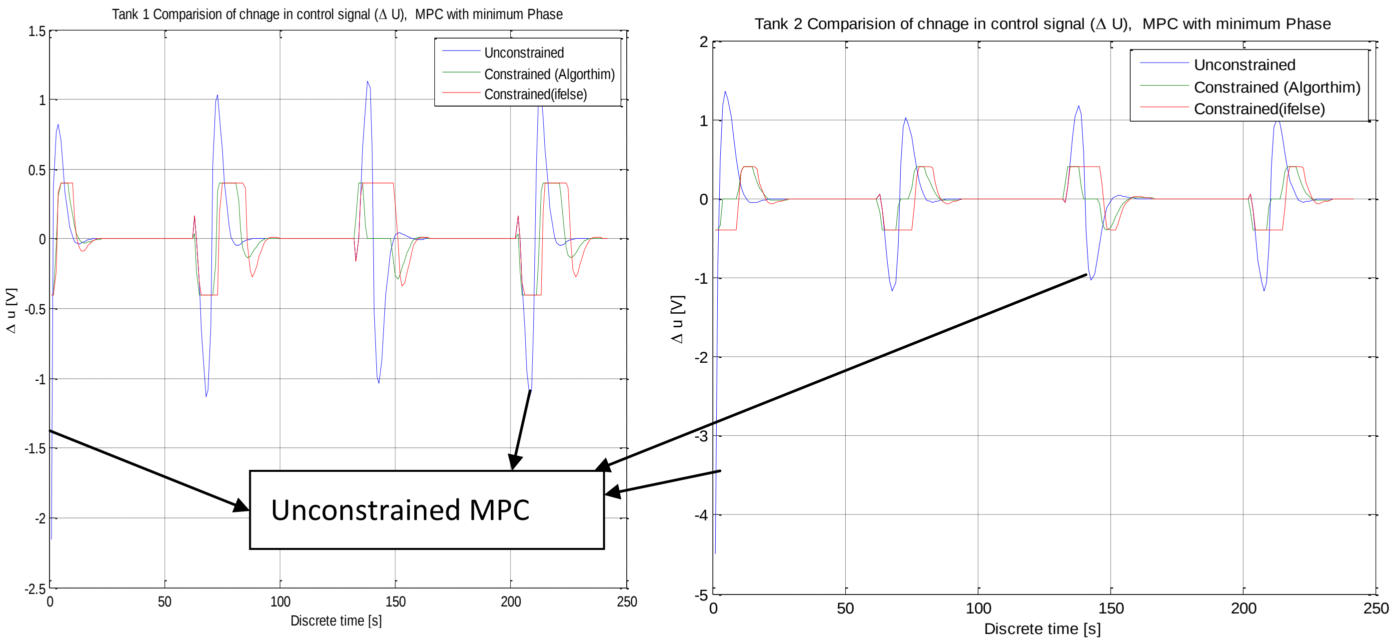


Figure 8-16: *Change in control signal U of Tank 1 and Tank 2 of minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action*

Referring to figure 8-16, which shows the plotting of change in control signal Δu, which is the voltage to pump is compared with different type of MPC. The constrained applied on change in control signal U is -0.04 $\leq \Delta u \leq$0.04, this value is chosen randomly. These constraints are valid for both tanks. But figure 8-16 shows the unconstrained MPC, provides a change in control signal maximum of nearly 1.2 volt and minimum of nearly -4.5 V, which violates the required constraints limit. It clearly shows that constrained MPC (algorithm and if else loop) follows the applied constraints, whenever unconstrained violated the constrained limit rise up to 1.2v, the change in control signal was 0.04v for constrained MPC. This clearly shows constrained MPC are following the constraints limit.

Remark 3: *Based on the reference and output plot for tank level, unconstrained MPC was much better controller as it followed the reference much quicker with less overshoot or undershoots. But referring to control signal plot and change in control signal plot, it was observed that unconstrained MPC violated the physical constraints limit, hence unconstrained MPC is not preferred controller for minimum phase quadruple tank system.*

By comparing the constrained MPC on algorithm based and if else loop, it is observed from control and change in control signal plot that both follow the required constraint limit. Based on the observation of reference and output plot for tank 1 and tank 2, it clearly shows that MPC constraint algorithm based (red colour in the plot) has less overshoot and follows the reference much quicker as compared to MPC constrained if else loop.

MPC constrains algorithm based follows the constraint limit and also follows the reference much quicker comparing with other MPC techniques.

8.2.2 Non-Minimum Phase

This section provides the comparison of various MPC techniques on non minimum phase quadruple tank process. Three different MATLAB code was simulated namely Unconstrained MPC, constrained MPC (Algorithm based) and Constrained MPC (if else loop), these data's were written to a excel file and plotted, coding provided on appendix 11.

Table 8-4, provides the list of critical parameter which were used in simulation of MPC controller, these values were common to different types of simulation. The matrices R and Q were selected on trial and error method.

Table 8-4: *Critical parameter for MPC simulation on non minimum phase quadruple tank process.*

N	***L***	***R***	***Q***	***Constraints on Control signal U (Pump voltage) [Volts]***				***Tank 1 Set point***	***Tank 2 Set point***
				U_{max}	U_{min}	ΔU_{max}	ΔU_{min}		
250	8	0.09*eye(2)	50*eye(2)	5	0	0.4	-0.4	14.1 & 13.9 cm	12.1 and 11.9 cm

Figure 8-17, provides a comparison plot of difference type of MPC simulation, where Tank 1 and tank 2 levels are compared with reference level. Different colour coding clearly distinguishes the type of controller. As all output change their state before the reference signals changes, hence all simulation follows MPC main behaviour. With reference to figure 8-17, it is clearly visible that unconstrained MPC follows the reference much faster than other types of controller and also it has less overshoots compared to others.

Hence from the figure 8-17, it is understood that unconstrained MPC settles/follows the reference much faster, than its constrained based MPC which follows the reference with a little delay. For tank 2, constrained MPC (algorithm and if else based) follows the reference with much delay, overshoot observed in tank 2 is similar for different type of MPC. Based on this plot Unconstrained MPC is preferable.

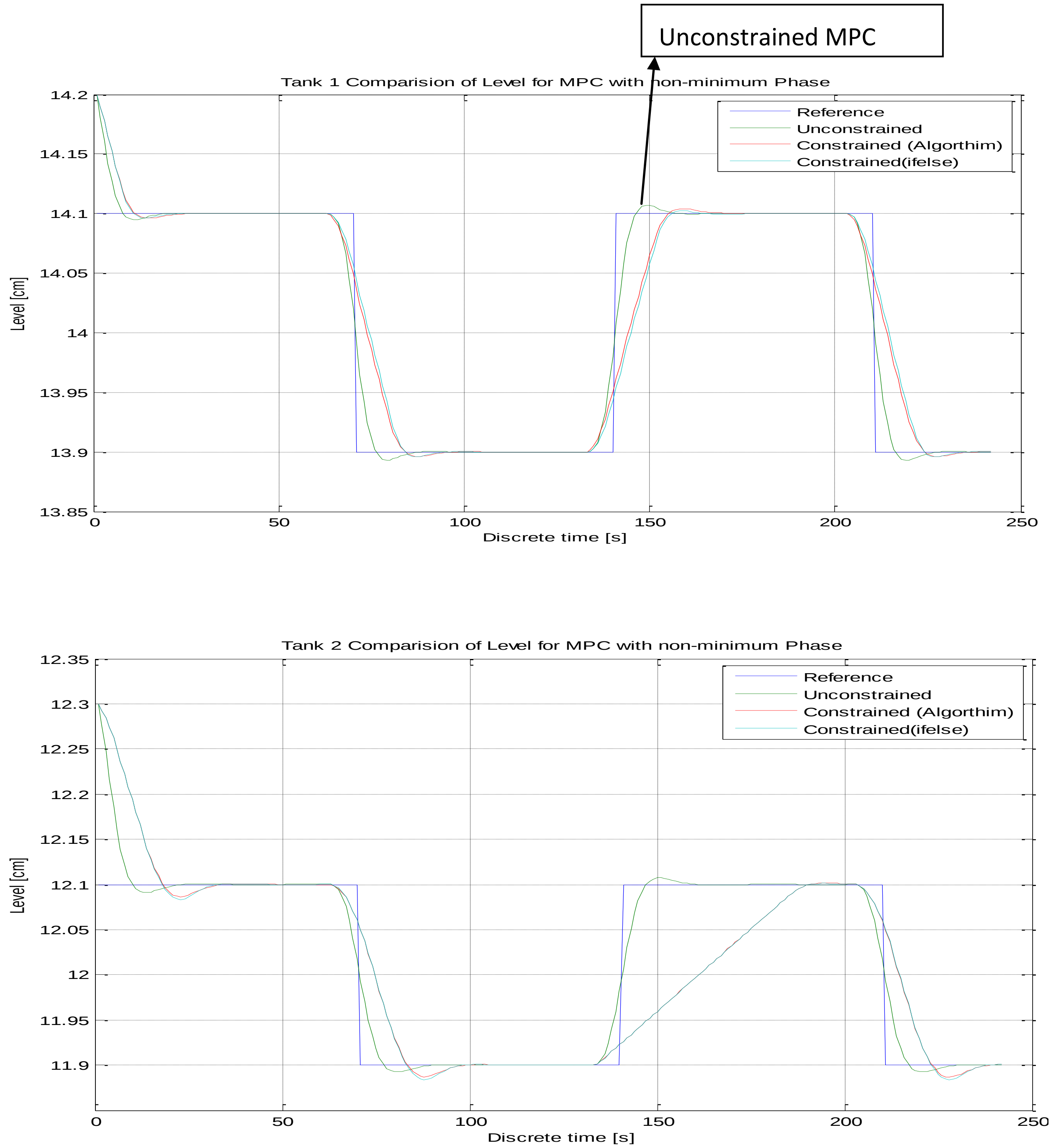


Figure 8-17: *Level of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action.*

Referring to plot 8-18, which shows the plotting of input control signal U, which is the voltage to pump is compared with different type of MPC. The constrained applied on control signal U is $0 \leq U \leq 5$, this is due to physical/mechanical constraint on instruments. This constraint is valid for both tanks. But the figure shows the unconstrained MPC, provides a control signal which higher than 5v and lower than 0v. For tank1, the higher value is nearly 8.2v and lower value is nearly -3.2v whereas for tank2, the higher value is 10.2v and lower value is nearly -4.2v, which clearly violates the constraint limits on control signal.

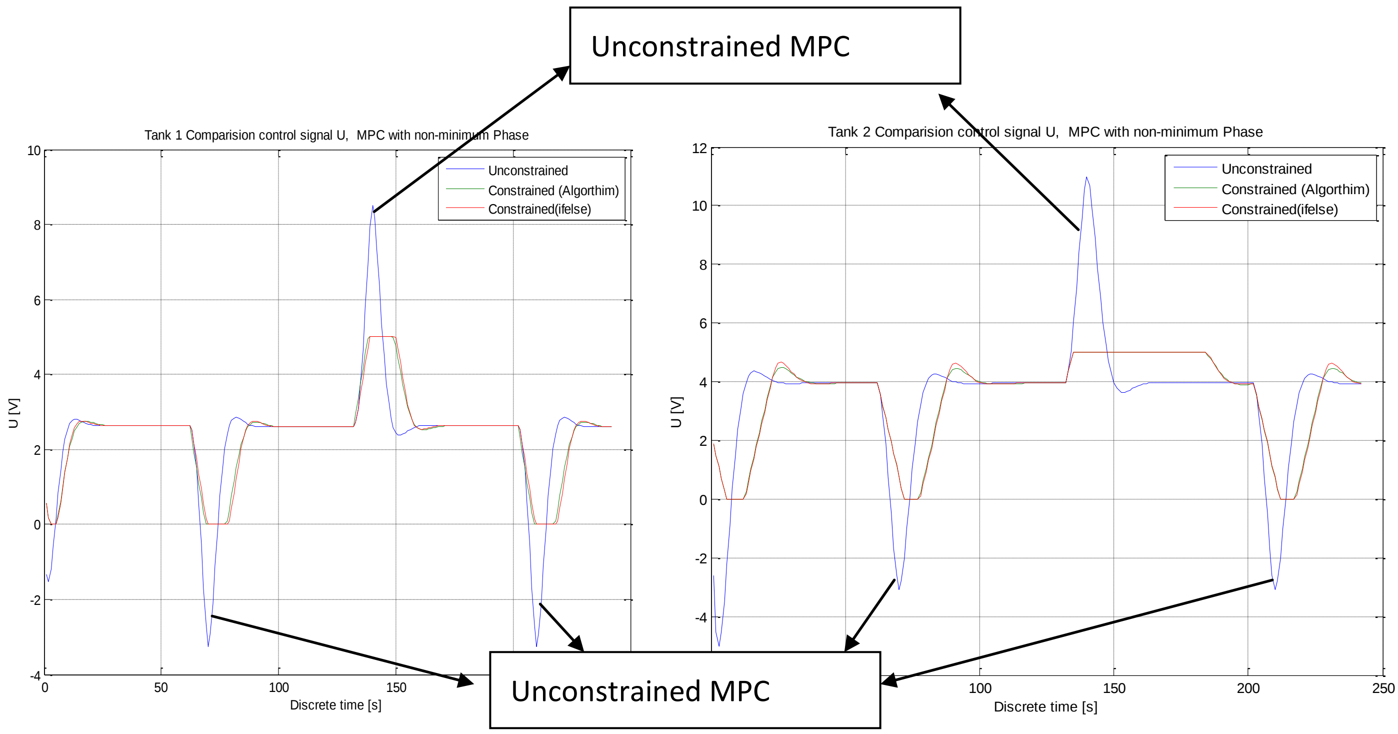


Figure 8-18: *Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action*

In the same figure 8-18, it clearly shows that constrained MPC (algorithm and if else loop) follows the applied constraints, whenever unconstrained violated the constrained limit and rise up to 10.2v, the control signal was 5v for constrained MPC. This clearly shows constrained MPC are following the constraints limits.

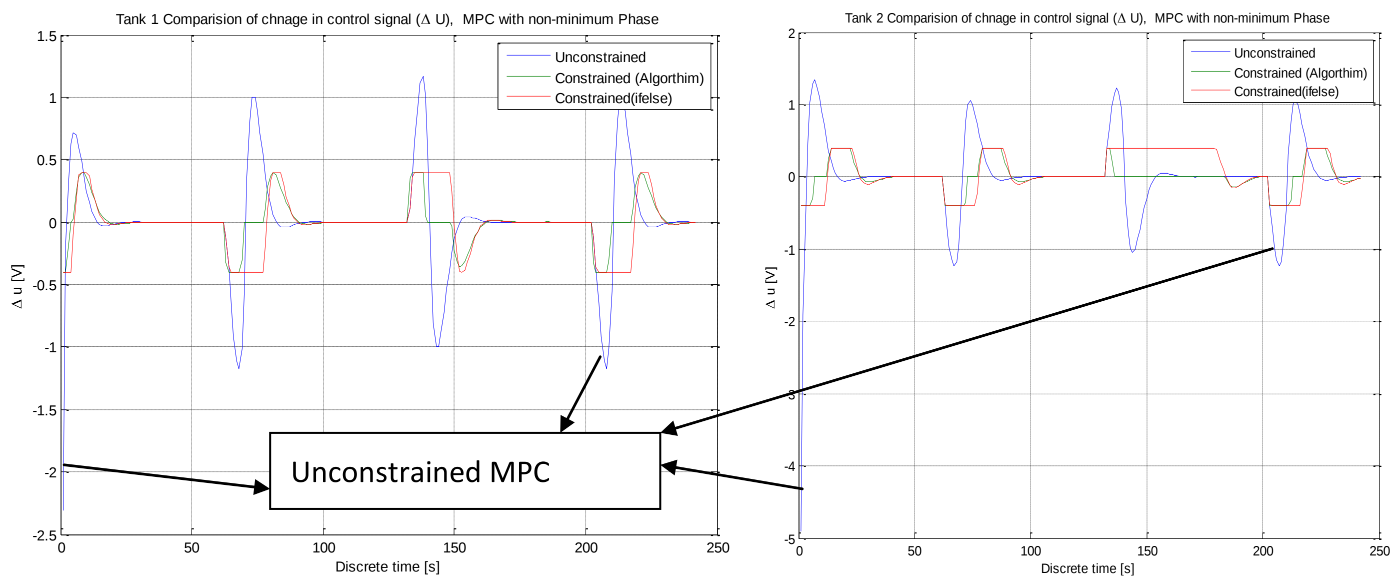


Figure 8-19: *Change in Control signal U of Tank 1 and Tank 2 of non minimum phase quadruple tank system comparison based on constraints and unconstrained MPC with integral action*

Referring to plot 8-19, which shows the plotting of change in control signal Δu, which is the voltage to pump is compared with different type of MPC. The constrained applied on change in control signal Δu is -0.04 $\leq \Delta u \leq$0.04, this value is chosen randomly. This constraint is valid for both tanks. But this figure shows the unconstrained MPC, provides a change in control signal maximum value of nearly 1.2 volt and minimum value of nearly -5 V, which doesn't follow the required constraints limit. It clearly shows that constrained MPC (algorithm and if else loop) follows the applied constraints, whenever unconstrained violated the constrained limit rise up to 1.2v, the change in control signal was 0.04v for constrained MPC. This clearly shows constrained MPC are following the constraints limits.

Remark 4: *Based on the reference and output plot for tank level, unconstrained MPC was much better controller as it followed the reference much quicker. But referring to control signal plot and change in control signal plot, it was observed that unconstrained MPC violated the physical constraints limit. Unconstrained MPC violated the control signal U and change in control signal* Δu *constraint limit, hence unconstrained MPC is not preferred controller for non-minimum quadruple tank system.*

By comparing the constrained MPC on algorithm based and if else loop, it is observed from control signal plot and change in control signal plot that both follow the required constraint limit. Based on the observation of reference and output plot for tank 1 and tank 2, it's observed for tank 1 that MPC constrained and MPC if else simulation nearly follows each other and settles at the set point at nearly same time. For tank 2, it's clearly visible that both simulations overlap each other.

MPC constrains algorithm and MPC if else loop both follows the constraint limit and also follows the reference much quicker comparing with other unconstrained MPC.

8.3 LQ, MPC and PI comparison

One of important requirement of this master thesis is comparison of various controllers on the basis of constraints handling on quadruple tank process. The performance of these controllers mainly depends on their algorithm and how these controllers handle constraints. The comparisons of all the controllers are based on minimum phase of the quadruple tank system.

Regarding LQ and MPC it's already discussed on how constraints are been handle, PI controller handle constraint through hard limit (if else), as it can't handle through algorithm. PI controller computes the input control signal U directly, hence constraints can be implemented on U only and for ΔU constraints can't be imposed as PI controller doesn't compute change in control signal. A separate MATLAB coding was executed for PI controller for minimum and non-minimum phase of quadruple tank process refer appendix 12.

In order to compare the LQ, MPC and PI controller, the simulation time 'N', should be same, hence N was chosen as 2500 and all programs were modified and simulated and results were transferred to a excel file and plotted. Refer table 8-5 which provides the constraint limits on $U\ and\ \Delta u$.

Table 8-5: *Constraint limits on all Controller (for PI only U constraints used)*

$\boldsymbol{U_{max}}$	$\boldsymbol{U_{min}}$	$\Delta\boldsymbol{U_{max}}$	$\Delta\boldsymbol{U_{min}}$
5	0	0.04	-0.04

Figure 8-20 shows the plots of various controllers performance or tank 1 level outputs with respect to reference signal. The reference was changed between 12.5 cm and 12.3 cm. The following points are observed based on figure 8-20.

- PI controller (pink colour in the plot) follows the reference signal at every change, but it takes long time to settle, whereas a LQ or MPC controller settles much quicker than PI controller. PI controller has big overshoot compared to others, reason for lager overshoot can be referred to figure 8-23 where the control signal U is larger compared to other controller for longer duration.
- LQ controller (green colour in the plot) follows the reference signal with perfection and no overshoots or undershoots are observed when there is a change in reference signal, it takes less time than a PI controller to settle. But LQ controller takes more time to settle compared to MPC constrained controller.
- MPC Algorithm based (red colour in plot) and MPC if else based (pale green colour) controller follows the reference at each instant of time. Both of these controllers follow the reference much quicker than any other controller. Even both these

controllers more or less have the same values or the same path they follow to the reference. But in order to find a difference between these two controllers, from figure 8-20, it's observed that MPC if else based constraint controller has a larger undershoot compared to MPC algorithm based controller.

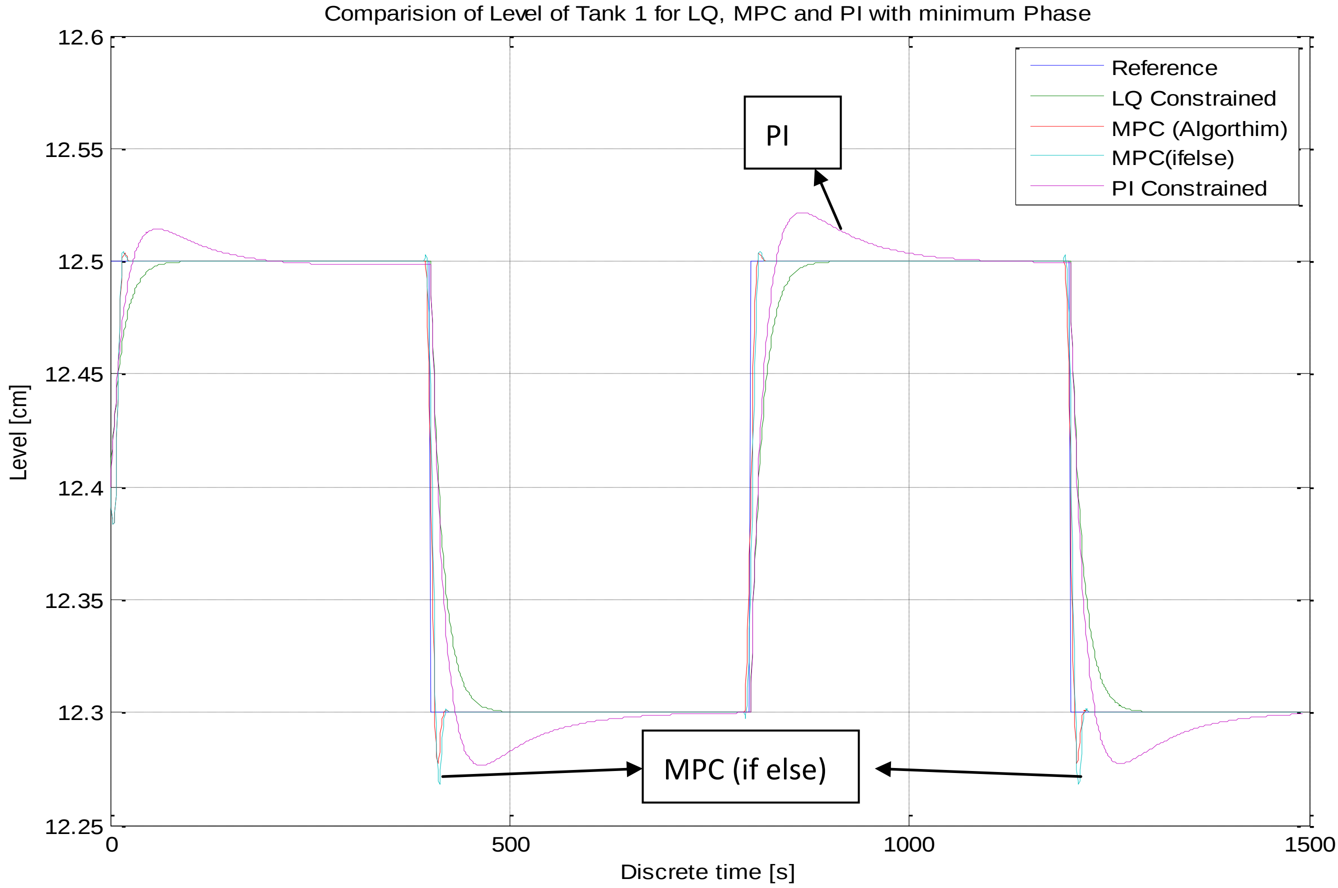


Figure 8-20: *Comparison of LQ, MPC and PI control with reference signal for Tank 1. Constrained implemented in all controller. It's a minimum phase quadruple tank process.*

Figure 8-21 shows the plot of various controllers performances or tank 2 level output with respect to reference signal. The reference was changed between 12.8 cm and 12.6 cm. The following points are observed based on figure 8-21.

- LQ controller (green colour in plot) follows the reference signal and attains the required setpoint. Comparing with others controller, it's observed from figure 8-21 that LQ controller takes more time to reach the setpoint/reference signal than other controllers. A point to be noticed is LQ controller has no overshoot or undershoots.
- PI controller (pink colour in the plot) follows the reference signal at every change, but it has a big undershoot and overshoots compared to other controllers. It takes less time than LQ controller but takes more time than MPC controller to reach the setpoint.
- MPC Algorithm based (red colour in plot) and MPC if else based (pale green colour) controller follows the reference at each instant of time. Both these controllers have

less overshoot compared to PI controller. Even both these controllers more or less have the same values or follow the same path to the reference. Even the overshoot values of both these controllers are more or less the same value. But if the picture is zoomed and observed, it clear that MPC if else based constraint has bigger undershoot compared to MPC algorithm based constraint.

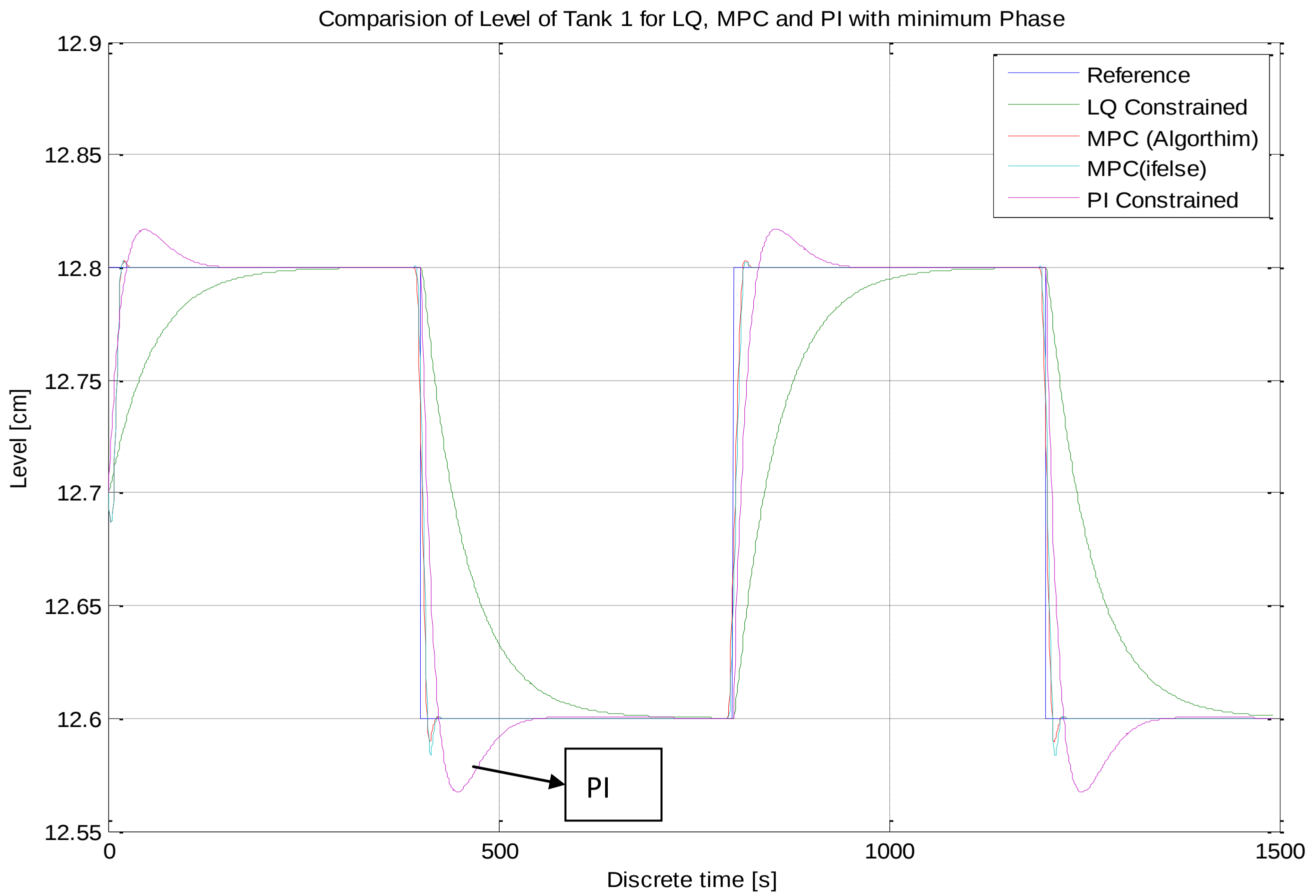


Figure 8-21: *Comparison of LQ, MPC and PI control with reference signal for Tank 2. Constrained implemented in all controller. It's a minimum phase quadruple tank process.*

To have a check on constraints whether all simulation were within the constrained limits, figure 8-22 and 8-23 shows the plotting of control signal U for every time stamp for various controller. Figure 8-22 shows tank 1 control signal and it's observed that U has the maximum value of 5 and minimum value of 0, hence constraints limit has been followed by all controllers. Similarly figure 8-23 shows tank 2 control signals, which also clearly shows that all values of control signal U are within the limit.

One interesting thing to be noted from figure 8-22 and 8-23 is that MPC algorithm based and MPC if else constraints often attain the peak value of 5v, when there is a change in setpoint in tank level. PI controller doesn't reach the peak value of 5v, but the voltage level reaches 4.5v when there is change in setpoint in tank level. PI controllers control signal varies rapidly, but these changes introduce overshoots in the plot and take a long time to settle with the reference signal. On the other hand LQ controller, most of the times its nearby 3v and very little changes in the control signal U value. These can one of the reasons for long settling time for LQ controller.

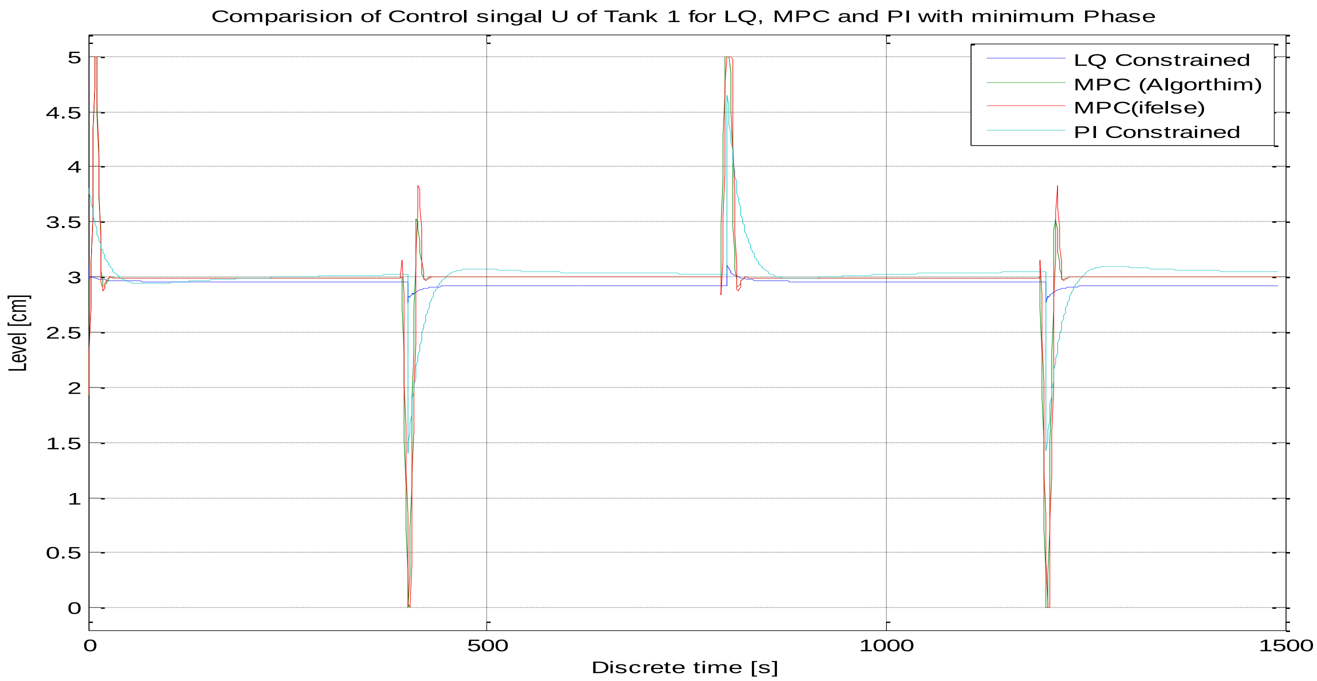


Figure 8-22: *Comparison of LQ, MPC and PI control, based on constraint limit (0≤U≤5) on control signal U for Tank 1. It's a minimum phase quadruple tank process.*

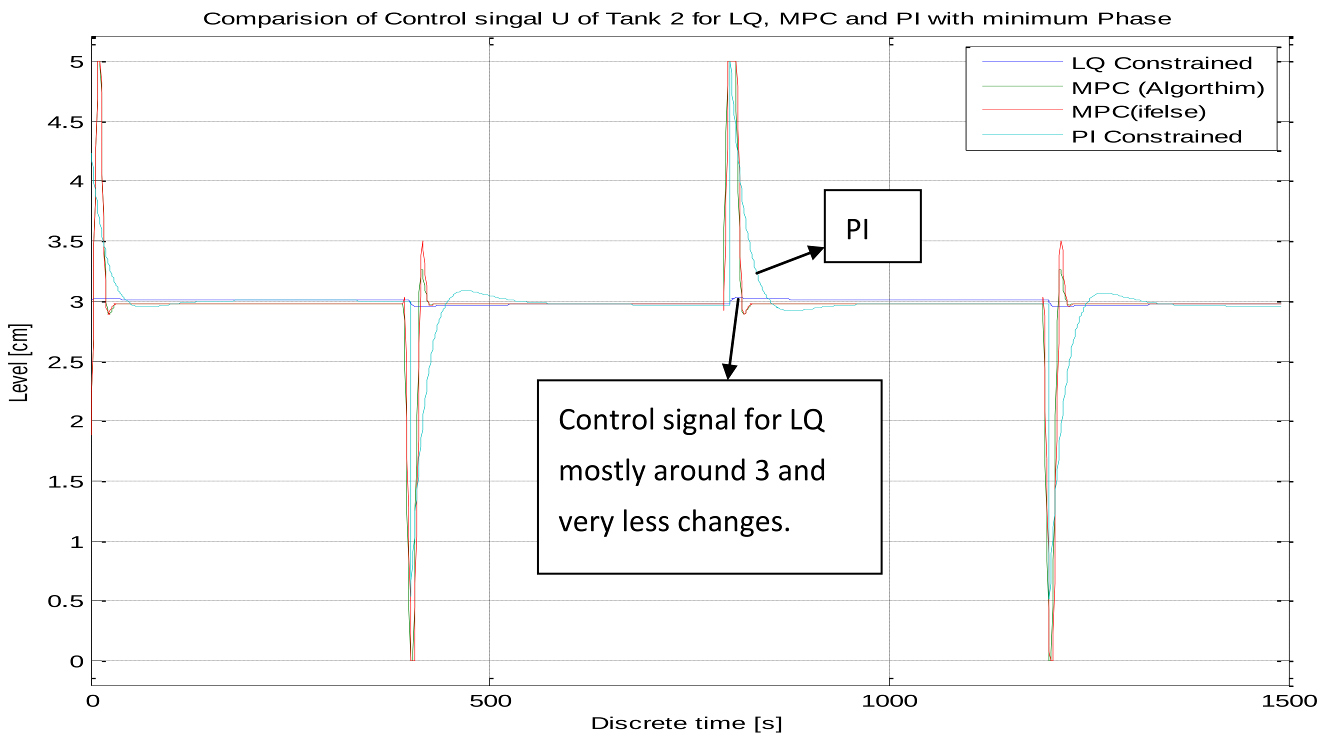


Figure 8-23: *Comparison of LQ, MPC and PI control, based on constraint limit (0≤U≤5) on control signal U for Tank 2. It's a minimum phase quadruple tank process*

Figure 8-24, provides the plotting of change in control signal ΔU for tank 1 and tank 2 with respect to time. The constraints are $-0.4 \leq \Delta U \leq 0.4$, from the figure its clear that all controller follows the constraints limit. As PI controller calculates only control signal U, hence change in control signal can't be plotted for PI control. From the figure it is observed that LQ optimal controller change in control signal ΔU values changes very little whereas MPC algorithm based and MPC if else loop controller change in control signal ΔU vary to the maximum and minimum values.

Table 8-6 provides a comparison of various controllers with constraints limitation on a minimum phase quadruple tank process. Based on these comparisons, it is apt to say that MPC algorithm based constraint controller is preferred for this process.

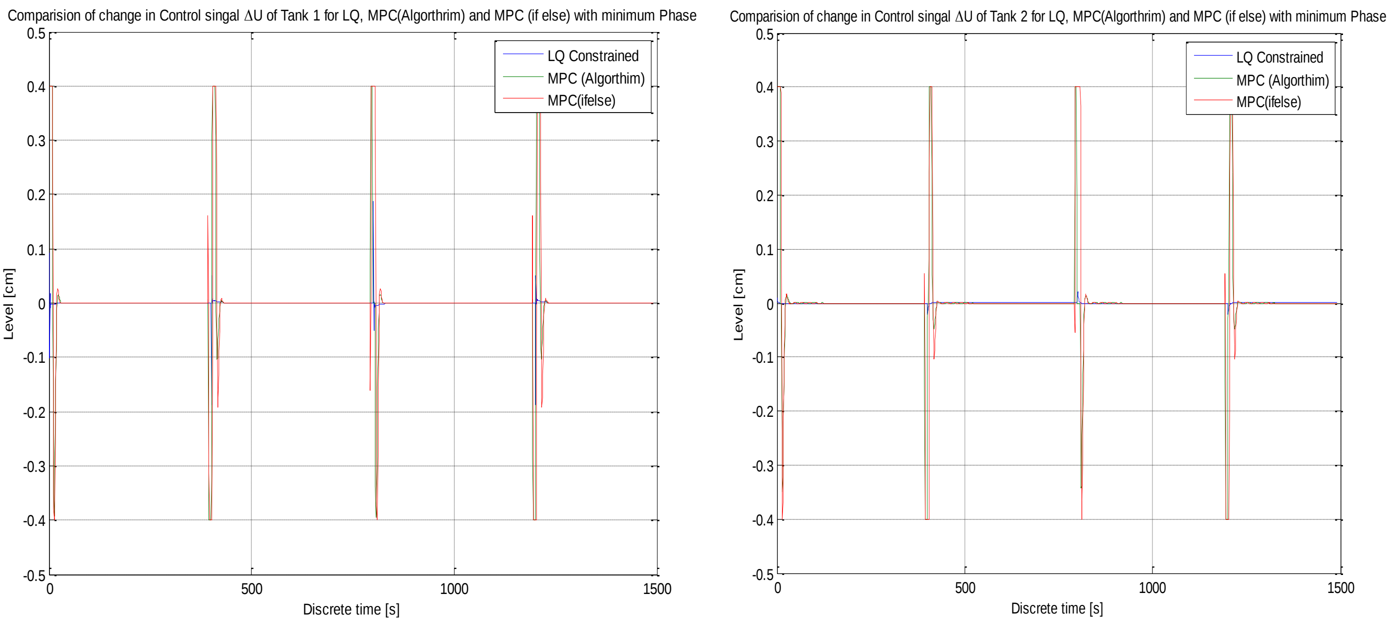


Figure 8-24: *Comparison of LQ, MPC and PI control, based on constraint limit (-0.4≤ΔU≤0.4) on control signal ΔU for tank 1 and tank 2. It's a minimum phase quadruple tank process.*

Table 8-6: *Comparison overview of various controllers based on constraint for minimum phase quadruple tank system.*

Controller (Constrained)	***Control Signal value observed (u) (Tank1, Tank2)***		***Change in Control Signal value observed (Δu) (Tank1, Tank2)***		***Overshoot***	***Remark***
	u_{max}	u_{min}	Δu_{max}	Δu_{min}		
MPC Algorithm based	5,5	0,0	0.4,0.4	-0.4,-0.4	Very less	Settles quicker than LQ and PI, but takes same time as MPC if else based. Less overshoot or undershoots than MPC if else loop. *Preferred Controller*.
MPC if else loop	5,5	0,0	0.4,0.4	-0.4,-0.4	Very less	Settles quicker than LQ and PI, but takes same time as MPC Algorithm based, but it has more overshoot and undershoots compared to MPC Algorithm.
LQ Optimal controller	3.1,3.02	2.77,2.95	0.18,0.02	-0.18,-0.02	No	Less time than PI but more time than MPC to settle. Very less change in control signal may be the reason for long time to settle. No overshoot or undershoot observed.
PI controller	4.5, 5	1.4,0.52	NA	NA	High	Takes very long time to settle and high control signal u for long duration (figure 8-22 and 8-23) makes larger overshoot.

9 Future Developments

This thesis provides the various results and simulation for the tasks mentioned in appendix 1. Improvements are always a part of scientific study. This thesis can also be improved or future developments can be made as follows

- MPC can handle constraints in its algorithm itself, similarly algorithm for PI and LQ shall be found where the constraints are handled in the algorithm itself.

- This thesis discusses results mostly from input/control signal and change in control signal constraints. Output constraints implementation can be considered as a future development or any other type of constraints should be also be implemented in the algorithm for any type of controller.

- Model of the tank was obtained from Johansson 2000. A new model of the four tank process can also be obtained using experiment analysis such as system identification.

- The operating point of four tank was also obtained from Johansson 2000, new operating point or the steady state value for four tank can be found from simulating the non linear model.

- Results are discussed based on simulation of the four tank process in MATLAB. The written program can be implemented in the real process and results can be compared.

- Simulation of the four tank process can also be executed in LABVIEW and results of MATLAB and LABVIEW can be compared.

10 Conclusion

Linear quadratic optimal control and Model predictive control with integral action was studied in detail in theory, equation and simulation. Multiple input multiple output, highly interactive system are hard to control, hence to compare these both controllers quadruple tank process was selected which is a non linear, MIMO , unknown slowly varying process, measurement disturbance and continuous process. Non linear model was converted to linear model.

Constraints handling in LQ optimal control and MPC was studied and discussed in detail. Simulations were performed on quadruple tank process based on LQ, MPC and PI controller with the main focus on constraints handling.

It is observed that LQ optimal control constrained (if else loop) and unconstrained take same time to reach the set point for minimum and non-minimum phase system the reason for this behaviour was mainly due to the very less change in control signal Δu. As change in control signal was very less, even the constrained LQ was within the constraint limits, hence same behaviour was observed. Unconstrained MPC controller reached the set point much quicker than other MPC control, but it violated the constraints limits for minimum and non minimum phase system. For minimum phase system algorithm based MPC control reaches the set point much quicker and has less overshoot compared to MPC with if else loop. For non-minimum phase the performance of MPC algorithm based and MPC if else are similar.

Various controllers comparison was performed based on constraints handling for minimum phase system and its concluded that MPC algorithm based constraints handling reaches the set point much quicker, MPC if else constraint handling reaches next but with little overshoot, LQ optimal control with if else constraints handling reaches later than MPC and finally PI controller with if else constraints takes long time to reach the set point.

It is also observed that MPC algorithms in the unconstrained form are equivalent to Linear quadratic control the major change was the mathematical algorithm and equations.

Based on the comparison it's concluded that Model predictive control with algorithm based constraint handling reaches the set point quicker and satisfies the constraint limit for quadruple tank process.

PI Controller and LQ optimal controller (state feedback controller) has exactly the same structure.

Constraints handling are apt and more preferable if it is accommodated in the algorithm itself, may be this is the reason behind the better performance of MPC with algorithm based constraints.

Reference

ARSLAN, G. 2005. 351M Digital Signal Processing.

BENAVOLI, A., L.CHISCI, FARINA, A., ORTENZI, L. & ZAPPA, G. Hard Constrainted vs Soft constraints parameter estimation. *IEEE Transcation on aerospace and electronic systems*.

D.VISWANATH 2011. Study of Minimum and Non-minimum Pahse systems.

ENCYLOPEDIA, W. T. F. 2012. *Constraints Optimization* [Online]. Internet. Available: http://en.wikipedia.org/wiki/Constraint_optimization [Accessed 22-April-2013 2013].

FINDEISEN., R. & ALLGOWER, F. An Introduction to Nonlinear Model Predictive Control. *21st Benelux Meeting on Systems and Control*.

GARCIA, M, P. D. & M, M. 1989. Model Predictive Control: Theory and Practive-a survery. 335-348.

HALVORSEN., H.-P. 2011. Model Predictive Control in Labview. 15.

HAUSER, K. 2012. B553 Lecture 7: Constrainted Optimization, Lagrange Multiplier and KKT Conditions. 12.

HOLKAR., K. S. & L.M.WAGHMARE 2010. An overview of Model Predictive Control. *International journal of control and Automation,* 3.

HUANG, Y., ZHANG, W. & ZHANG, H. 2006. Infinite Horizon LQ optimal control for discrete time stochastics systems. *Congress on Intelligent control and automation*.

JOHANSSON, K. H. May 2000. The Quadruple-Tank process: A multivariable Laboratory process with an adjustable zero. *IEEE Transcation on control system technology,* 8**,** 10.

LANCASTER, P. & RODMAN, L. 1995. Algebraic Riccati equations.

LEE., J.-W., KWON, W. H. & CHOI, J. On Stability of Constrainted Receding Horizon control with finite terminal weighting matrix.

MORARI, M. 2008. Predicting the Future of Model Predictive Control.

MORDUKHOVICH, B. S. & SHVARSMAN, I. 2012. Approximate Maximum Principle for Discrete oprimal control system.

NATIONAL, I. 2012. Labview Help.

NICOLA, G. D., MAGNI, L. & SCATTOLINI, R. 2000. Stability and Robutness of Nonlinear Receding Horizon Control.

NIKOLAOU, M. 2011. Model Predictive Controllers: A critical synthesis of theory and Industrila Needs. 1-2.

NUNES, G. C. 2001. Desgin and Analysis of Multivariable predictive control applied to an oil-water gas seperator: A polynomial approach. 126.

QIN, S. J. & BADGWELL, T. A. An Overview of Industrial MPC.

RAWLINGS, J. B., MEADOWS, E. S. & MUSKE, K. R. 1994. Non-Linear Model Predictive Control: A tutorial and survey.

RICHTER, S. 2010. IfA Fachpraktikum-Experiment 3.4: Quad Tanks. 25.

RICKER, N. L. 1991. Model Predictive Control: State of the art. 271-296.

RUSCIO, D. D. 1996. System Theory State space analysis and Control theory. 162.

RUSCIO, D. D. 2012a. Discrete LQ Optimal Control Integral Action: A simple controller on incremental form for MIMO system. *Norwegian Soceity of Automation*.

RUSCIO, D. D. 2012b. Model Predictive Control and Optimization. 207.

RUSCIO, D. D. 2012c. Optimal Model Based Control: System Analysis and Desgin. 229.

TAKATSU, H., ITOH, T. & ARAKI, M. 1998. Future needs for the control theory in industries -report and topics of the control technology survey in Japanese industries. *Process control,* 8.

TODOROV, E. 2006. Optimal Control theory. *MIT Journal*.

TRIRATANAJARU, A. S. 2011. PID Controller Theory and Loop tuning.

UNIVERSITY, C. *Online* [Online]. Available: http://www.statslab.cam.ac.uk/~rrw1/oc/L13.pdf.

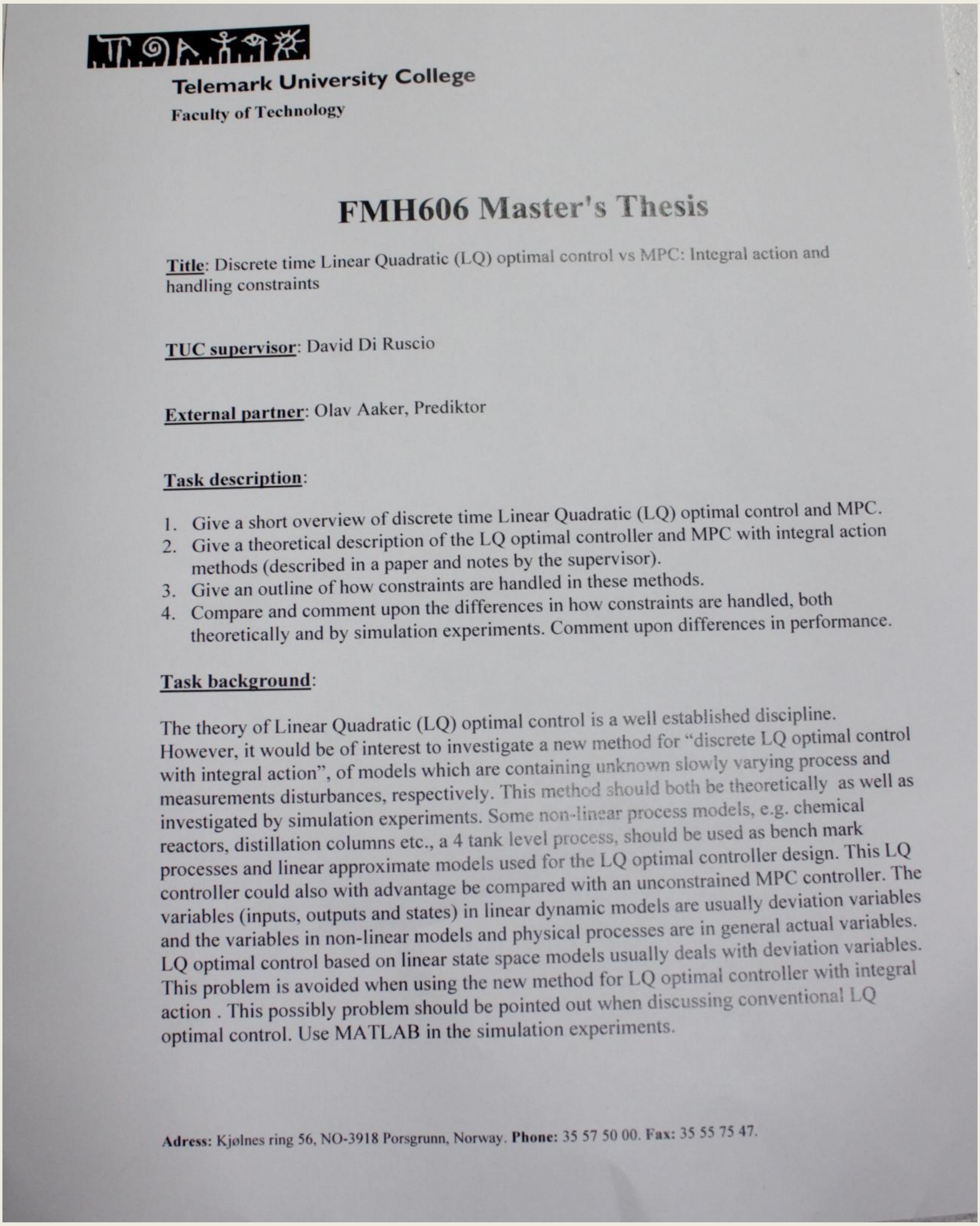
URGEN, K. & SCHOOR, G. V. 2011. Predicitive PID control of Non-minimum Phase systems. *Intech*.

VERHAEGEN, M., VERDULT, V. & BERGBOER, N. 2007. Filtering and System Identification: An Introduction to using Matlab Software.

WANG, L. 2009. Model Predictive Control system Desgin and Implemtation Using MATLAB. 403.

WELCH, G. & BISHOP, G. An Introduction to the kalman filter. 12.

Appendix 1: Master Thesis task description SIV-53-13



Telemark University College

Faculty of Technology

FMH606 Master's Thesis

Title: Discrete time Linear Quadratic (LQ) optimal control vs MPC: Integral action and handling constraints

TUC supervisor: David Di Ruscio

External partner: Olav Aaker, Prediktor

Task description:

1. Give a short overview of discrete time Linear Quadratic (LQ) optimal control and MPC.
2. Give a theoretical description of the LQ optimal controller and MPC with integral action methods (described in a paper and notes by the supervisor).
3. Give an outline of how constraints are handled in these methods.
4. Compare and comment upon the differences in how constraints are handled, both theoretically and by simulation experiments. Comment upon differences in performance.

Task background:

The theory of Linear Quadratic (LQ) optimal control is a well established discipline. However, it would be of interest to investigate a new method for "discrete LQ optimal control with integral action", of models which are containing unknown slowly varying process and measurements disturbances, respectively. This method should both be theoretically as well as investigated by simulation experiments. Some non-linear process models, e.g. chemical reactors, distillation columns etc., a 4 tank level process, should be used as bench mark processes and linear approximate models used for the LQ optimal controller design. This LQ controller could also with advantage be compared with an unconstrained MPC controller. The variables (inputs, outputs and states) in linear dynamic models are usually deviation variables and the variables in non-linear models and physical processes are in general actual variables. LQ optimal control based on linear state space models usually deals with deviation variables. This problem is avoided when using the new method for LQ optimal controller with integral action . This possibly problem should be pointed out when discussing conventional LQ optimal control. Use MATLAB in the simulation experiments.

Adress: Kjølnes ring 56, NO-3918 Porsgrunn, Norway. **Phone:** 35 57 50 00. **Fax:** 35 55 75 47.

Student category:

SCE students

Practical arrangements:

The work with the thesis will be held at TUC

Signatures:

Student (date and signature): Anna 28/Jan/2013

Supervisor (date and signature): 28/1 - 13 David DiRuscio

Appendix 2: Properties of Linearized Model

MATLAB code for analysis the properties of the Linearized model of quadruple tank system for minimum and non-minimum phase.

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 10-Feb-2013
%% MATLAB file to check the properties of the Linearized model
% of quadruple tank process

clear all;
clc;close all;


% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm2
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % volts/cm
g=981;% cm/sec2

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nProperties check for Linearized SSM of Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
Option=input('\nYour Choice:');


if Option==1 % minimum phase chosen

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.8;
h4=1.40;

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
```

```
y1=0.70;
y2=0.60;


elseif Option==2 % Non-minimum phase

% initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, the parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;

end

%Calculation of time constants
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;

% Model Development, This is the linear Model based on
% xdot = Ax +Bu
% y=Dx
% the below model is Continues Time model
A11=[-1/T1,0,A3/(A1*T3),0;
    0,-1/T2,0,A4/(A2*T4);
    0,0,-1/T3,0;
    0,0,0,-1/T4];

B1=[y1*k1/A1,0;
   0,y2*k2/A2;
   0,(1-y2)*k2/A3;
  (1-y1)*k1/A4,0];

D1=[kc,0,0,0;
   0,kc,0,0];


% Properties of the Linearized model SSM of quadruple tank system
Eigen_values_of_linearized_system=eig(A11)
controllanility_linear=ctrb(A11,B1);
obsvervability_linear=obsv(A11,D1);

% showing the properties on screen
Rank_of_Controllability_matrix=rank(controllanility_linear)
Rank_of_observability_matrix=rank(obsvervability_linear)
Transmission_zeroes_of_system_is_given_by=tzero(A11,B1,D1,zeros(2,2))
```

Appendix 3: LQ with Unconstrained

MATLAB coding for LQ Optimal control with unconstrained for Quadruple tank system

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 15-March-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with LQ Control with Integral action

clc
clear all; close all

% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % Pump gain [V/cm]
g=981; % Gravity [cm/s^2]

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nLQ Control of 4 tank system \n1 : Minimum Phase System ')
fprintf('\n2 : Non - Minimum Phase System ')
Option=input('\nYour Choice:');


if Option==1 % Minimum phase case

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.80;
h4=1.40;

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
y1=0.7;
y2=0.6;

% Initial pump voltages in Volts
```

```
u=[3;3];
Rw=10*[1,0;0,1];
Q=0.1*[100,0;0,1];


elseif Option==2 % Non-minimum phase case

%initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;

% Initial pump voltages in Volts
u=[3.15;3.15];

% LQ setting depending upon Min and NM phase
Rw=[.1,0;0,.1];
Q=[.01,0;0,.001];
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;

% Model Development, This is the linear Model based on
% xdot = Ax +Bu
% y=Dx
% the below model is Continues Time model
A=[-1/T1, 0 , A3/(A1*T3), 0
0 , -1/T2, 0 , A4/(A2*T4)
0 , 0 , -1/T3 , 0
0 , 0 , 0 , -1/T4];
B=[y1*k1/A1 ,0
0 ,y2*k2/A2
0 ,(1-y2)*k2/A3
(1-y1)*k1/A4,0];
D=[kc, 0, 0, 0
0, kc, 0, 0];

 %Discretzing the model (sampling time in sec.)
ts=0.1;
[Ac,Bc,Dc]=c2dm(A,B,D,zeros(2),ts,'zoh');

% Step length and time interval parameters
h=1; t0=0; t1=1500; N=(t1-t0)/h;
```

```
[G1,G2]=dlqdu_pi(Ac,Bc,Dc,Q,Rw); % LQ-controller
r_init=[h1,h2]; % Nominal reference for output y=kc*[h10;h20]

% Initial States
x=[h1;h2;h3;h4];
x_est=x;x_old=x;xx_old=x_old;
y_old=D*x_old; yy=y_old; yy_old=yy;
u_old=u;uu_old=u_old;
r=r_init;

% Calculation of Kalman Gain
G=eye(4);
Q1=10*eye(4);
R1=0.01*eye(2);
[Ke,Pp,Pe,E]=dlqe(A,G,D,Q1,R1);% given the covariance of the noise,
                               % kalman gain Ke can found


% Generating a random set point (Reference Signal rk)
% from a predefined m file made by David Di Ruscio
rand('seed',0), randn('seed',0)
ref=[h1*ones(N,1)+0.1*prbs1(N,400,400) ...
h2*ones(N,1)+0.1*prbs1(N,400,400)];

%Control Loop
for i=1:N
    y=D*x_est;
    ychec=D*x;
    r=ref;

    %Kalman filter Algorithm
    xp=x_old; % set initial (apiroi) predicted state estimate
    yp=D*xp;  % measurement update model
    ep=y-yp;  % estimator error
    xp=x_old+Ke*ep; % Corrected state estimate
    x_est=[xp(1);xp(2);xp(3);xp(4)];

if Option==1
du=G1*(x_est-x_old)+G2*(y_old-r(i,:)'); % finding delta U control signal
u=u_old+du; % u signal from kalman estimator

% Storing variables for next loop usage
x_old=x_est;
y_old=y;
u_old=u;

U(i,:)=u'; Y(i,:)=y'; R(i,:)=r(i,:); Y1(i,:)=ychec';
X_est(:,i)=x_est; XX(:,i)=x;DeltaU_x(i,:)=du';

%Non linear model simulation with estimated states
f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x_est=x_est+h*f;
```

```
%Non linear model simulation with measured states
f(1)=(-a1*sqrt(2*g*x(1)) +a3*sqrt(2*g*x(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x(2)) +a4*sqrt(2*g*x(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x=x+h*f;
end


if Option==2 %Non-Minimal Phase
du=G1*(x-x_old)+G2*(y_old-r(i,:)');
u=u_old+du;

x_old=x;
y_old=y;
u_old=u;

U(i,:)=u'; Y(i,:)=y'; R(i,:)=r(i,:);
X_est(:,i)=x_est; XX(:,i)=x;DeltaU_x(i,:)=du';

%Non linear model simulation with estimated states
f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x_est=x_est+h*f;

%Non linear model simulation with measured states
f(1)=(-a1*sqrt(2*g*x(1)) +a3*sqrt(2*g*x(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x(2)) +a4*sqrt(2*g*x(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x=x+h*f;
end
t0=t0+h;
end


T=1:t1; % Time stamp for plotting

% Plotting all details of Tank 1
figure(1)
subplot(222), plot(T,U(:,1)), grid on
title('Tank 1: Input Control signal to Pump')
ylabel('Input [V]'),xlabel('Time [s]')

subplot(221), plot(T,R(:,1),'r--',T,Y(:,1),'b-')
grid on
title('Tank 1: Output Level Vs Reference Plot'), ylabel('level [cm]')
legend('Reference','Output Level'), xlabel('Time [s]')
if Option==1
axis([0 1500 12.1 12.6])
else
axis([0 1500 12.2 12.8])
end
```

```
subplot(223),plot(T,X_est(1,:),'b')
hold on
grid on
plot(T,XX(1,:),'r'),title('Tank 1: Measured Level Vs Estimated Level Plot')
legend('Level Est.','Level Meas.'),xlabel('Time [s]')
if Option==1
axis([0 1500 11.5 13])
else
axis([0 1500 12.2 12.8])
end

subplot(224)
plot(DeltaU_x(:,1)), grid on
title('Tank 1: \DeltaU' )
xlabel('Time [s]'), ylabel('\DeltaU [V]')

% Plotting all details of Tank 2
figure(2)

subplot(222), plot(T,U(:,2)), grid on
title('Tank 2: Input Control signal to Pump')
ylabel('Input [V]'),xlabel('Time [s]')

subplot(221), plot(T,R(:,2),'r--',T,Y(:,2),'b-')
grid on
title('Tank 2: Output Level Vs Reference Plot'), ylabel('level [cm]')
legend('Reference','Output Level'), xlabel('Time [s]')
if Option==1
    axis([0 1500 12.4 13])
else
    axis([0 1500 12.7 13.5])
end

subplot(223),plot(T,X_est(2,:),'b',T,XX(2,:),'r'), grid on
title('Tank 2: Measured Level Vs Estimated Level Plot')
legend('Level Est.','Level Meas.'),xlabel('Time [s]'),ylabel('level [cm]')

subplot(224)
plot(DeltaU_x(:,2)),grid on
title('Tank 2: \DeltaU ')
xlabel('Time [s]'), ylabel('\DeltaU [V]')

figure (3) % tank 3 and 4 estimated level
subplot(211),plot(T,X_est(3,:)),title('Tank 3: Estimated Level')
grid on
xlabel('Time [s]')

subplot(212),plot(T,X_est(4,:)),title('Tank 4: Estimated Level')
grid on
xlabel('Time [s]')
```

Appendix 4: LQ Constrained (if else loop)

MATLAB coding for LQ Optimal control with constrained (if else) for Quadruple tank system

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 10-March-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with LQ Control with Integral action (Using Constrained Algorithm)
% if else constraints

clc
clear all;close all

% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % Pump gain [V/cm]
g=981; % Gravity [cm/s^2]

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nLQ Control (Constrained) of 4 tank system \n1 : Minimum Phase
System ')
fprintf('\n2 : Non - Minimum Phase System ')
Option=input('\nYour Choice:');


if Option==1 % Minimum phase case

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.80;
h4=1.40;

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
y1=0.7;
y2=0.6;
```

```
% Initial pump voltages in Volts
u=[3;3];
Rw=10*[1,0;0,1];
Q=0.1*[100,0;0,1];

elseif Option==2% Non-minimum phase case

%initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;

% Initial pump voltages in Volts
u=[3.15;3.15];

% LQ setting depending upon Min and NM phase
Rw=[.1,0;0,.1];
Q=[.01,0;0,.001];
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;

% Model Development, This is the linear Model based on
% xdot = Ax +Bu
% y=Dx
% the below model is Continues Time model
A=[-1/T1, 0 , A3/(A1*T3), 0
0 , -1/T2, 0 , A4/(A2*T4)
0 , 0 , -1/T3 , 0
0 , 0 , 0 , -1/T4];

B=[y1*k1/A1 ,0
0 ,y2*k2/A2
0 ,(1-y2)*k2/A3
(1-y1)*k1/A4,0];

D=[kc, 0, 0, 0
0, kc, 0, 0];

% Discretizing the model, Convert from continuous- to discrete-time model
ts=0.1;
[Ac,Bc,Dc]=c2dm(A,B,D,zeros(2),ts,'zoh');
```

```
% Step length and time interval parameters
h=1; t0=0; t1=1500; N=(t1-t0)/h;

% Constraints limit for control signal U
umax=5;umin=0;
delumax=0.4;delumin=-0.4;


[G1,G2]=dlqdu_pi(Ac,Bc,Dc,Q,Rw); % LQ-controller matrix
r_init=[h1,h2]; % Nominal reference for output y=kc*[h10;h20]

% Initial States
x=[h1;h2;h3;h4];
x_est=x;x_old=x;xx_old=x_old;
y_old=D*x_old; yy=y_old; yy_old=yy;
u_old=u;uu_old=u_old;
r=r_init;

% Calculation of Kalman Gain
G=eye(4);
Q1=10*eye(4);
R1=0.01*eye(2);
[Ke,Pp,Pe,E]=dlqe(A,G,D,Q1,R1); % given the covariance of the noise,
                                % kalman gain Ke can found

% Generating a random set point (Reference Signal rk)
% from a predefined m file made by David Di Ruscio
rand('seed',0), randn('seed',0)
ref=[h1*ones(N,1)+0.1*prbs1(N,400,400) ...
h2*ones(N,1)+0.1*prbs1(N,400,400)];

%Control Loop
for i=1:N
    y=D*x_est;
    r=ref;

    %Kalman filter Algorithm
    xp=x_old; % set initial (apiroi) predicted state estimate
    yp=D*xp;  % measurement update model
    ep=y-yp;  % estimator error
    xp=x_old+Ke*ep; % Corrected state estimate
    x_est=[xp(1);xp(2);xp(3);xp(4)];

    if Option==1
      du=G1*(x_est-x_old)+G2*(y_old-r(i,:)'); % finding delta U control
signal

      % Implementing constraints for delta U for first Input to Tank 1
      if du(1,1)>delumax
         du(1,1)=delumax;
      elseif du(1,1)<delumin
         du(1,1)=delumin;
      end

      %Implementing constraints for delta U for first Input to Tank 2
      if du(2,1)>delumax
         du(2,1)=delumax;
      elseif du(2,1)<delumin
```

```
         du(2,1)=delumin;
      end

    u=u_old+du; % New Control signal

    if u(1,1)>umax % implementing constraints for first Input signal U for
tank1
       u(1,1)=umax;
    elseif u(1,1)<umin
       u(1,1)=umin;
    end

    if u(2,1)>umax  % implementing constraints for first Input signal U for
tank2
       u(2,1)=umax;
    elseif u(2,1)<umin
       u(2,1)=umin;
    end

% Storing variables for next loop usage
x_old=x_est;
y_old=y;
u_old=u;

U(i,:)=u'; Y(i,:)=y'; R(i,:)=r(i,:);
X_est(:,i)=x_est; XX(:,i)=x;DeltaU_x(i,:)=du';

% Non linear model simulation with estimated states or predicted state
% estimate update, last step of kalman filter (step 4)
f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x_est=x_est+h*f;

% updation based on x measure value
f(1)=(-a1*sqrt(2*g*x(1)) +a3*sqrt(2*g*x(3)) +y1*k1*u(1))/A1;
f(2)=(-a2*sqrt(2*g*x(2)) +a4*sqrt(2*g*x(4)) +y2*k2*u(2))/A2;
f(3)=(-a3*sqrt(2*g*x(3)) + (1-y2)*k2*u(2))/A3;
f(4)=(-a4*sqrt(2*g*x(4)) + (1-y1)*k1*u(1))/A4;
f=[f(1);f(2);f(3);f(4)];
x=x+h*f;
end



if Option==2 %Non-Minimal Phase
du=G1*(x-x_old)+G2*(y_old-r(i,:)'); % finding delta U control signal

     %Implementing constraints for delta U for first Input to Tank 1
     if du(1,1)>delumax
        du(1,1)=delumax;
     elseif du(1,1)<delumin
        du(1,1)=delumin;
     end

     %Implementing constraints for delta U for first Input to Tank 2
     if du(2,1)>delumax
```

```
            du(2,1)=delumax;
         elseif du(2,1)<delumin
            du(2,1)=delumin;
         end

     u=u_old+du; % New Control signal

     % implementing constraints for first Input signal U for tank
        if u(1,1)>umax
           u(1,1)=umax;
        elseif u(1,1)<umin
           u(1,1)=umin;
        end

     % implementing constraints for first Input signal U for tank2
        if u(2,1)>umax
           u(2,1)=umax;
        elseif u(2,1)<umin
           u(2,1)=umin;
        end
     % Storing variables for next loop usage
     x_old=x;y_old=y;u_old=u;uu_old=u;

     U(i,:)=u'; Y(i,:)=y'; R(i,:)=r(i,:);
     X_est(:,i)=x_est; XX(:,i)=x;DeltaU_x(i,:)=du';

     %Non linear model simulation with estimated states
     f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
     f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
     f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
     f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
     f=[f(1);f(2);f(3);f(4)];
     x_est=x_est+h*f;

     %Non linear model simulation with measured states
     f(1)=(-a1*sqrt(2*g*x(1)) +a3*sqrt(2*g*x(3)) +y1*k1*u(1))/A1;
     f(2)=(-a2*sqrt(2*g*x(2)) +a4*sqrt(2*g*x(4)) +y2*k2*u(2))/A2;
     f(3)=(-a3*sqrt(2*g*x(3)) + (1-y2)*k2*u(2))/A3;
     f(4)=(-a4*sqrt(2*g*x(4)) + (1-y1)*k1*u(1))/A4;
     f=[f(1);f(2);f(3);f(4)];
     x=x+h*f;
     end
     end

     T=1:t1; % Time stamp for plotting

     % Plotting the simulation results

     % Plotting all details of Tank 1
     figure(1)
     subplot(221), plot(T,R(:,1),'r--',T,Y(:,1),'b-')
     grid on
     title('Tank 1: Output Level Vs Reference Plot')
     if Option==1
         axis([0 1500 12.2 12.6])
     end
     ylabel('level [cm]'),legend('Reference','Output Level'), xlabel('Time [s]')

     subplot(222), plot(T,U(:,1)), grid on
```

```
title('Tank 1: Input Control signal with Constraints')
ylabel('U [V]'),xlabel('Time [s]')
legend('Umax:5 Umin:0')

subplot(223),plot(T,X_est(1,:),'b')
hold on ,grid on
plot(T,XX(1,:),'r')
title('Tank 1: Measured Level Vs Estimated Level Plot')
if Option==1
    axis([0 1500 11.2 13])
end
ylabel('level [cm]')
legend('Level Est.','Level Meas.'),xlabel('Time [s]')

subplot(224)
plot(DeltaU_x(:,1)), grid on
title('Tank 1: \DeltaU with Constraints')
xlabel('Time [s]'), ylabel('\DeltaU [V]')
legend('\DeltaUmax:0.4 \DeltaUmin:-0.4')

% Plotting all details of Tank 2
figure(2)
subplot(221), plot(T,R(:,2),'r--',T,Y(:,2),'b-')
grid on
title('Tank 2: Output Level Vs Reference Plot')
ylabel('level [cm]'),legend('Reference','Output Level'), xlabel('Time [s]')
if Option==1
    axis([0 1000 12.4 13])
else
    axis([0 1000 12.7 13.5])
end

subplot(222), plot(T,U(:,2)), grid on
title('Tank 2: Input Control signal with Constraints')
ylabel('U [V]'),xlabel('Time [s]')
legend('Umax:5 Umin:0')

subplot(223),plot(T,X_est(2,:),'b',T,XX(2,:),'r')
title('Tank 2: Measured Level Vs Estimated Level Plot')
ylabel('level [cm]'), grid on
legend('Level Est.','Level Meas.'),xlabel('Time [s]')

subplot(224)
plot(DeltaU_x(:,2)),grid on
title('Tank 2: \DeltaU with Constraints')
xlabel('Time [s]'), ylabel('\DeltaU [V]')
legend('\DeltaUmax:0.4 \DeltaUmin:-0.4')

% Plotting Tank 3 and Tank 4 Estimated level
figure (3)
subplot(211),plot(T,X_est(3,:)), legend('Level Est.')
title('Tank 3: Estimated Level'), ylabel('Level [cm]')
grid on
xlabel('Time [s]')
subplot(212),plot(T,X_est(4,:)), legend('Level Est.')
title('Tank 4: Estimated Level'), ylabel('Level [cm]')
grid on
xlabel('Time [s]')
```

Appendix 5: Writing & Reading data from Excel File (Minimum Phase)

MATLAB coding for LQ Optimal minimum phase system, writing and reading from excel file

Writing to excel file:

```
% for first program for LQ minimum phase
xlswrite('C:\Users\Annamalai\Google Drive\Thesis\code\final\LQmin.xlsx',ref)
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',Y(:,1),'c1:c1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',Y(:,2),'d1:d1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',U(:,1),'e1:e1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',U(:,2),'f1:f1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',DeltaU(:,1),'g1:g1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',DeltaU(:,2),'h1:h1500')

% for second program for constrained LQ using if else minimum phase
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',Y(:,1),'i1:i1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',Y(:,2),'j1:j1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',U(:,1),'k1:k1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',U(:,2),'l1:l1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',DeltaU(:,1),'m1:m1500')
xlswrite('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx',DeltaU(:,2),'n1:n1500')
```

Reading from excel file and plotting

```
%% MATLAB file to plot and compare the various parameter for Consrained
% and unconstrained LQ with minimum phase system

clc
close all
t=1:1500;

% Tank 1 , Output Level comparision
ref_tank1=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','a1:a1500');
Y_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','c1:c1500');
Y_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','i1:i1500');

figure(1)
plot(t,[ref_tank1,Y_tank1_Uncon,Y_tank1_conalgo]);
```

```
legend('Reference','Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 1 Comparision of Level for LQ with minimum Phase')
grid on

% Tank 2  , Output Level comparision
ref_tank2=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','b1:b1500');
Y_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','d1:d1500');
Y_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','j1:j1500');

figure(2)
plot(t,[ref_tank2,Y_tank2_Uncon,Y_tank2_conalgo]);
legend('Reference','Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 2 Comparison of Level for LQ with minimum Phase')
grid on

% Tank 1 , Control signal comparison
U_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','e1:e1500');
U_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','k1:k1500');
figure(3)
plot(t,[U_tank1_Uncon,U_tank1_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 1 Comparision control signal U,  LQ with minimum Phase')
grid on

% Tank 2 , Control signal comparison
U_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','f1:f1500');
U_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','l1:l1500');
figure(4)
plot(t,[U_tank2_Uncon,U_tank2_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 2 Comparision control signal U,  LQ with minimum Phase')
grid on

% Tank 1 , change in Control (delta U)signal comparison
delU_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','g1:g1500');
delU_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','m1:m1500');
figure(5)
plot(t,[delU_tank1_Uncon,delU_tank1_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 1 Comparison of change in control signal (\Delta U),  LQ with
minimum Phase')
grid on

% Tank 2 , change in Control (delta U)signal comparison
delU_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','h1:h1500');
```

```
delU_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQmin.xlsx','n1:n1500');
figure(6)
plot(t,[delU_tank2_Uncon,delU_tank2_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 2 Comparison of change in control signal (\Delta U),  LQ with
minimum Phase')
grid on
```

Appendix 6: Writing & Reading data from Excel File (Non-Minimum Phase)

MATLAB coding for LQ Optimal non minimum phase system, reading from excel file and plotting.

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
LQ
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to plot and compare the various parameter for LQ with
% non-minimum phase system

clc
close all
t=1:1500;

% Tank 1 , Output Level comparison
ref_tank1=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','a1:a1500');
Y_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','c1:c1500');
Y_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','i1:i1500');

figure(1)
plot(t,[ref_tank1,Y_tank1_Uncon,Y_tank1_conalgo]);
legend('Reference','Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 1 Comparision of Level for LQ with non-minimum Phase')
grid on

% Tank 2  , Output Level comparison
ref_tank2=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','b1:b1500');
Y_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','d1:d1500');
Y_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','j1:j1500');
```

```
figure(2)
plot(t,[ref_tank2,Y_tank2_Uncon,Y_tank2_conalgo]);
legend('Reference','Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 2 Comparison of Level for LQ with non-minimum Phase')
grid on

% Tank 1 , Control signal comparison
U_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','e1:e1500');
U_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','k1:k1500');
figure(3)
plot(t,[U_tank1_Uncon,U_tank1_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 1 Comparision control signal U,  LQ with non-minimum Phase')
grid on

% Tank 2 , Control signal comparison
U_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','f1:f1500');
U_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','l1:l1500');
figure(4)
plot(t,[U_tank2_Uncon,U_tank2_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 2 Comparision control signal U,  LQ with non-minimum Phase')
grid on

% Tank 1 , change in Control (delta U)signal comparison
delU_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','g1:g1500');
delU_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','m1:m1500');
figure(5)
plot(t,[delU_tank1_Uncon,delU_tank1_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 1 Comparison of change in control signal (\Delta U),  LQ with
non-minimum Phase')
grid on

% Tank 2 , change in Control (delta U)signal comparison
delU_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','h1:h1500');
delU_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\LQnonmin.xlsx','n1:n1500');
figure(6)
plot(t,[delU_tank2_Uncon,delU_tank2_conalgo]);
legend('Unconstrained','Constrained (ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 2 Comparison of change in control signal (\Delta U),  LQ with
non-minimum Phase')
grid on
```

Appendix 7: MPC constrained (Algorithm based)

MATLAB code for MPC constrained (Algorithm based) with integral action

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with MPC Control with Integral action (Using Constrained Algorithm)
% quadprog constraints

clear all;
clc;close all;

% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm2
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % volts/cm
g=981;% cm/sec2

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nMPC (Constraints) Control of Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
Option=input('\nYour Choice:');

if Option==1 % minimum phase chosen

% initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.8;
h4=1.40;
x0=[h1;h2;h3;h4];

% Initial pump voltages in Volts
u10=3;
u20=3;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
```

```
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
y1=0.70;
y2=0.60;

elseif Option==2 % Non-minimum phase

% initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;
x0=[h1;h2;h3;h4];

% Initial pump voltages in Volts
u10=3.15;
u20=3.15;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;
TimeConstantOfFourTank=[T1 T2 T3 T4];

% Model Development, This is the linear Model based on
% xdot = Ax +Bu
% y=Dx
% the below model is Continues Time model
A=[-1/T1,0,A3/(A1*T3),0;
    0,-1/T2,0,A4/(A2*T4);
    0,0,-1/T3,0;
    0,0,0,-1/T4];

B=[y1*k1/A1,0;
   0,y2*k2/A2;
   0,(1-y2)*k2/A3;
   (1-y1)*k1/A4,0];

D=[kc,0,0,0;
   0,kc,0,0];

% Discretizing the model, Convert from continuous- to discrete-time model
ts=0.1;%sampling time in sec, either 0.1 or 1 used
[A,B,D]=c2dm(A,B,D,zeros(2),ts,'zoh'); %  discretization
```

```
% Reading the size of the Discrete model, these size are used to create
% augmented model
n=size(A,1); % Reading the number of Rows, State variable
r=size(B,2); % Reading the number of Columns, Inputs
m=size(D,1); %Reading the number of Rows, Outputs


%Augmented Model Development or Extended state space model in terms of
%deviation variable
At=[A zeros(n,m); D eye(m,m)];
Bt=[B;zeros(m,r)];
Dt=[D eye(m,m)];

L=8; %Prediction horizon

%Weighting matrices
Q=50*eye(m);    % Output Error weight matrix
                % As per the size of output, 'm' is the size or number of
                % inputs

R=0.09*eye(r); % rate of change in control action weight matrix
                % As per the size of input, 'r' is the size or the number of
                % inputs


% Calculating parameters need for MPC like HdL, OL and OLB (Toeplitz matrix
% and Observability matrix) from a  predefined m file called ss2h
[HdL,OL,OLB]=ss2h(At,Bt,Dt,zeros(m,r),L,0); % Note sending the extended
                                             % state space model
FL=[OLB HdL];

% Make extended weight matrix based on length of Prediction Horizon
% This m file is made by David Di Ruscio
Qt=q2qt(Q,L);
Rt=q2qt(R,L);

% Variables need for Constrained MPC u(k,L) = S du(k,L) + c u(k-1)
[S,c] = scmat(r,L);
umax=5;
umin=0;
delumax=0.4;
delumin=-0.4;

% Varaibles need for Constrained MPC
H=FL'*Qt*FL+Rt;

% Simulation Horizon
N=250;

% Generating a random set point (Reference Signal rk)
% from a predefined m file made by David Di Ruscio
rand('seed',0);
ref=[14*ones(N,1)+0.1*prbs1(N,70,70) 12*ones(N,1)+0.1*prbs1(N,70,70)];
% based on this reference signal, the initial height is also taken near to
% it.

% Calculation of Kalman Gain
G=eye(4);
Q1=10*eye(4);
```

```
R1=0.01*eye(2);
[Ke, Pp,Pe,E]=dlqe(A,G,D,Q1,R1); % given the covariance of the noise,
                                 % kalman gain Ke can found


% Initial States
r1=[14.2;12.3]; % initial height of tank 1 and tank 2 in cm
Hd=D*inv(eye(4)-A)*B;
us=inv(Hd)*r1;
xs=inv(eye(4)-A)*B*us; % Initial height of all four tank
x=xs;xold=x;
x_est=xold; %For kalman filter initial estimated value to be old value
u=us;uold=u;
yold=D*x;

%Control Loop
for k=1:N-L
    y=D*x_est; % Height of Tank 1 and tank 2 based on measured value of x
               % this value is obtained to use in kalman filter estimation
               % error and out value of height in tank 1 and tank 2.

    % Kalman filter Algorithm
    xp=xold; % set initial (apiroi) predicted state estimate
    yp=D*xp; % measurement update model
    ep=y-yp; % estimator error
    xp=xold+Ke*ep; % Corrected state estimate
    x_est=[xp(1);xp(2);xp(3);xp(4)];

    % Make the extended reference vector based on the number of prediction
    % horizion
    rf =ref(k+1:k+L,:);
    ref_predhori=rf(1,:)';
    for i=2:L
        ref_predhori=[ref_predhori;rf(i,:)'];
    end

    % Compute MPC control
    xt=[x_est-xold;yold];
    pl=OL*At*xt;
    f=FL'*Qt*(pl-ref_predhori);

    % Constrained MPC Control Using quadprog along with input constraints
    b=[umax*ones(L*r,1)-c*uold;-umin*ones(L*r,1)+c*uold];
    a=[S;-S];
    %b=[delumax*ones(L*r,1);delumin*ones(L*r,1);umax*ones(L*r,1)-c*uold;-
umin*ones(L*r,1)+c*uold];
    %a=[eye(16);-eye(16);S;-S];
    duf=quadprog(H,f,a,b);

    % Implementing constraints for delta U for first Input to Tank 1
    if duf(1,1)>delumax
    duf(1,1)=delumax;
    elseif duf(1,1)<delumin
    duf(1,1)=delumin;
    end

    % Implementing constraints for delta U for first Input to Tank 2
    if duf(2,1)>delumax
    duf(2,1)=delumax;
```

```
    elseif duf(2,1)<delumin
    duf(2,1)=delumin;
    end


    u=u+duf(1:r);% Computing the new control signal for 2 tanks
    uold=u; % transferring the current ctrl to old ctrl signal for next
loop

    % Storing variables for plotting
    Y(k,:)=y'; U(k,:)=u';DeltaU(k,:)=duf(1:r)';X_est(:,k)=x_est;XX(:,k)=x;

    % Storing the current values as old values for next loop
    xold=x_est;
    yold=y;

    % Non linear model simulation with estimated states or predicited state
    % estimate update , last step of kalman filter (step 4)
    f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
    f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
    f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
    f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
    f=[f(1);f(2);f(3);f(4)];
    x_est=x_est+ts*f;

    % updation based on x measure value
    f1=((-a1*sqrt(2*g*x(1)))/A1)+(a3*(sqrt(2*g*x(3)))/A1)+((y1*k1*u(1))/A1);
    f2=((-a2*sqrt(2*g*x(2)))/A2)+(a4*(sqrt(2*g*x(4)))/A2)+((y2*k2*u(2))/A2);
    f3=((-a3*sqrt(2*g*x(3)))/A3)+(((1-y2)*k2*u(2))/A3);
    f4=((-a4*sqrt(2*g*x(4)))/A4)+(((1-y1)*k1*u(1))/A4);
    f=[f1;f2;f3;f4];
    x=x+ts*f;
end


t=1:N-L;  % Time stamp for plotting

figure(1) % Plotting the result for tank 1
plot(t,[Y(:,1) ref(1:N-L,1)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time [s]')
title('MPC Constrained Simulation: Output level y_k and reference r_k for
tank 1')
legend('y_k','r_k')

figure(2)% Plotting the result for tank 2
plot(t,[Y(:,2) ref(1:N-L,2)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time[s]');
legend('y_k','r_k');
title('MPC Constrained Simulation: Output level y_k and reference r_k for
tank 2')

figure(3) % plotting Control Signals of U for both tanks
subplot(211), plot(t,U(1:242,1)), grid on, ylabel('u1_k [V]')
axis([0 250 -0.1 5.1])
legend('Umax: 5 and Umin: 0');
title('Tank 1: control input, u_k')
subplot(212), plot(t,U(1:242,2)), grid on, ylabel('u2_k [V]')
xlabel('Discrete time [s]')
title('Tank 2: control input, u_k')
```

```
axis([0 250 -0.1 5.1])
legend('Umax: 5 and Umin: 0');


figure(4) % plotting Control Signals for delta U for both tanks
subplot(211), plot(t,DeltaU(:,1)), grid on, ylabel('deltau1_k [V]')
axis([0 250 -0.55 0.9])
legend('DeltaUmax: 0.4 and DeltaUmin: -0.4');
title('Tank 1: Delta U control signal ')
subplot(212), plot(t,DeltaU(:,2)), grid on, ylabel('deltau2_k [V]')
xlabel('Discrete time [s]')
title('Tank 2: Delta U control signal ')
axis([0 250 -0.55 0.9])
legend('DeltaUmax: 0.4 and DeltaUmin: -0.4');

figure(5) % Plot of Estimated and Measured levels
subplot(221),plot(t,X_est(1,:),'b'),grid on, hold on
plot(t,XX(1,:),'r'),title('Tank 1'), xlabel('Time[s]'),ylabel('Level [cm]')
legend('Level Est.','Level Meas.')
subplot(222),plot(t,X_est(2,:),'b',t,XX(2,:),'r'),title('Tank 2')
grid on
legend('Level Est.','Level Meas.'), xlabel('Time[s]'),ylabel('Level [cm]')
subplot(223),plot(t,X_est(3,:)),title('Tank 3'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
subplot(224),plot(t,X_est(4,:)),title('Tank 4'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
```

Appendix 8: MPC constrained (if else loop)

MATLAB code for MPC constrained (if else loop) with integral action

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with MPC Control with Integral action (Using Constrained Algorithm)
% If-else constraints

clear all;
clc; close all;

% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in c
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1;%volts/cm
g=981;% cm/sec2

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nMPC (Constraints) Control of Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
Option=input('\nYour Choice:');

if Option==1 % minimum phase

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.8;
h4=1.40;
x0=[h1;h2;h3;h4];

% initial pump voltages in Volts
u10=3;
u20=3;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
```

```
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
y1=0.70;
y2=0.60;

elseif Option==2 % Non-minimum phase

% initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;
x0=[h1;h2;h3;h4];

% Initial pump voltages in Volts
u10=3.15;
u20=3.15;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;
TimeConstantOfFourTank=[T1 T2 T3 T4];

%Model Development, This is the linear Model based on
%Hdot = Ac x +Bc u
%y=Dc x
%the below model is Continues Time model

A=[-1/T1,0,A3/(A1*T3),0;
    0,-1/T2,0,A4/(A2*T4);
    0,0,-1/T3,0;
    0,0,0,-1/T4];

B=[y1*k1/A1,0;
    0,y2*k2/A2;
    0,(1-y2)*k2/A3;
    (1-y1)*k1/A4,0];

D=[kc,0,0,0;
    0,kc,0,0];

% Discritizing the model, Convert from continuous- to discrete-time model
ts=0.1;%sampling time in sec, either 0.1 or 1 used
fprintf('\n The Discrete Model of the Non Linear Four tank Process')
```

```
[A,B,D]=c2dm(A,B,D,zeros(2),ts,'zoh'); %  discretization

% Reading the size of the Discrete model, these size are used to create
% augmented model,
n=size(A,1); % Reading the number of Rows, State variable
r=size(B,2); % Reading the number of Columns, Inputs
m=size(D,1); %Reading the number of Rows, Outputs


% Augmented Model Development or Extended state space model in terms of
% deviation variable
At=[A zeros(n,m); D eye(m,m)];
Bt=[B;zeros(m,r)];
Dt=[D eye(m,m)];

L=8; %Prediction horizon

% Weighting matrices
Q=100*eye(m); % Output Error weight matrix
              % As per the size of output, 'm' is the size or number of
              % inputs

R=0.1*eye(r); % rate of change in control action weight matrix
              % As per the size of input, 'r' is the size or the number of
              % inputs

% Calculation parameters need for MPC like HdL, OL and OLB (Toeplitz matrix
% and Observability matrix) from a predefined m file called ss2h
[HdL,OL,OLB]=ss2h(At,Bt,Dt,zeros(m,r),L,0); % Note sending the extended
                                            %state space model
FL=[OLB HdL];

% Make extended weight matrix based on length of Prediction Horizon
% This m file is made by David Di Ruscio
Qt=q2qt(Q,L);
Rt=q2qt(R,L);

% limits for the maximum and minimum value of Control Signal U and delta U
% Constraints of U (Control Signal)
umax=5;
umin=0;
delumax=0.4;
delumin=-0.4;

% Computing variables need for Unconstrained MPC
H=FL'*Qt*FL+Rt;

% Simulation Horizon
N=250;

% Generating a random set point (Reference Signal rk)
% from a predefined m file made by David Di Ruscio
rand('seed',0);
ref=[14*ones(N,1)+0.1*prbs1(N,70,70) 12*ones(N,1)+0.1*prbs1(N,70,70)];
% based on this reference signal, the initial height is also taken near to
% it.

%Parameters for Calculating Kalman gain
G=eye(4);
```

```
Q1=10*eye(4);
R1=0.01*eye(2);
[Ke,Pp,Pe,E]=dlqe(A,G,D,Q1,R1); % given the covariance of the noise,
                                % kalman gain Ke can found

% Initial States
r1=[14.2;12.3]; % initial height of tank 1 and tank 2 in cm
Hd=D*inv(eye(4)-A)*B;
us=inv(Hd)*r1;
xs=inv(eye(4)-A)*B*us; % Initial height of all four tank
x=xs;xold=x;
u=us;uold=u;
x_est=xold; %For kalman filter initial estimated value to be old value
yold=D*x;

%Control Loop
for k=1:N-L
    y=D*x_est; % Height of Tank 1 and tank 2 based on measured value of x
               % this value is obtained to use in kalman filter estimation
               % error and out value of height in tank 1 and tank 2.
    % Kalman filter Algorithm
    xp=xold; % set initial (apiroi) predicted state estimate
    yp=D*xp; % measurement update model
    ep=y-yp; % estimator error
    xp=xold+Ke*ep; % Corrected state estimate
    x_est=[xp(1);xp(2);xp(3);xp(4)];

    % Make the extended reference vector based on the number of prediction
    % horizon
    rf =ref(k+1:k+L,:);
    ref_predhori=rf(1,:)';
    for i=2:L
        ref_predhori=[ref_predhori;rf(i,:)'];
    end

    % Compute MPC control
    xt=[x_est-xold;yold];
    pl=OL*At*xt;
    f=FL'*Qt*(pl-ref_predhori);

    % finding delta U
    duf=-inv(H)*f;

    % if-else for implementing constraints
    % Implementing constraints for delta U for first Input to Tank 1
    if duf(1,1)>delumax
    duf(1,1)=delumax;
    elseif duf(1,1)<delumin
    duf(1,1)=delumin;
    end

    % Implementing constraints for delta U for first Input to Tank 2
    if duf(2,1)>delumax
    duf(2,1)=delumax;
    elseif duf(2,1)<delumin
    duf(2,1)=delumin;
    end

    u=u+duf(1:r); % Computing the new control signal for 2 tanks
```

```
    uold=u; % transferring the current ctrl to old ctrl signal for next
loop

    % Implementing Constraints using if else loop for U for tank 1
    if u(1,1)>umax
    u(1,1)=umax;
    elseif u(1,1)<umin
    u(1,1)=umin;
    end

    % implementing constraints for first Input signal U for tank2
    if u(2,1)>umax
    u(2,1)=umax;
    elseif u(2,1)<umin
    u(2,1)=umin;
    end

    % Storing variables for plotting
    Y(k,:)=y'; U(k,:)=u';X(k,:)=x'; DeltaU(k,:)=duf(1:r)';
X_est(:,k)=x_est;XX(:,k)=x;

   % Storing the current values as old values for next loop
    xold=x_est;
    yold=y;


   % Non linear model simulation with estimated states or predicted state
   % estimate update, last step of kalman filter (step 4)
   f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
   f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
   f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
   f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
   f=[f(1);f(2);f(3);f(4)];
   x_est=x_est+ts*f;

   % updation based on x measure value
   f1=((-a1*sqrt(2*g*x(1)))/A1)+(a3*(sqrt(2*g*x(3)))/A1)+((y1*k1*u(1))/A1);
   f2=((-a2*sqrt(2*g*x(2)))/A2)+(a4*(sqrt(2*g*x(4)))/A2)+((y2*k2*u(2))/A2);
   f3=((-a3*sqrt(2*g*x(3)))/A3)+(((1-y2)*k2*u(2))/A3);
   f4=((-a4*sqrt(2*g*x(4)))/A4)+(((1-y1)*k1*u(1))/A4);
   f=[f1;f2;f3;f4];
   x=x+ts*f;
end


t=1:N-L;  % Time stamp for plotting

figure(1) % Plotting the result for tank 1
plot(t,[Y(:,1) ref(1:N-L,1)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time [s]')
title('MPC Constrained Simulation: Output level y_k and reference r_k for
tank 1')
legend('y_k','r_k')

figure(2)%Plotting the result for tank 2
plot(t,[Y(:,2) ref(1:N-L,2)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time[s]');
legend('y_k','r_k');
```

```
title('MPC Constrained Simulation: Output level y_k and reference r_k for
tank 2')

figure(3) % plotting Control Signals of U for both tanks
subplot(211), plot(t,U(:,1)), grid on, ylabel('u1_k [V]')
axis([0 200 -0.1 5.1])
legend('Umax: 5 and Umin: 0');
title('Tank 1: control input, u_k')
subplot(212), plot(t,U(:,2)), grid on, ylabel('u2_k [V]')
xlabel('Discrete time [s]')
axis([0 200 -0.1 5.1])
title('Tank 2: control input, u_k')
legend('Umax: 5 and Umin: 0');


figure(4) % plotting Control Signals for delta U for both tanks
subplot(211), plot(t,DeltaU(:,1)), grid on, ylabel('deltau1_k [V]')
axis([0 200 -0.55 0.9])
legend('DeltaUmax: 0.4 and DeltaUmin: -0.4');
title('Tank 1: Delta U control signal ')
subplot(212), plot(t,DeltaU(:,2)), grid on, ylabel('deltau2_k [V]')
xlabel('Discrete time [s]')
axis([0 200 -0.55 0.9])
title('Tank 2: Delta U control signal ')
legend('DeltaUmax: 0.4 and DeltaUmin: -0.4');



figure(5) % Plot of Estimated and Measured levels
subplot(221),plot(t,X_est(1,:),'b'),grid on, hold on
plot(t,XX(1,:),'r'),title('Tank 1'), xlabel('Time[s]'),ylabel('Level [cm]')
legend('Level Est.','Level Meas.')
subplot(222),plot(t,X_est(2,:),'b',t,XX(2,:),'r'),title('Tank 2')
grid on
legend('Level Est.','Level Meas.'), xlabel('Time[s]'),ylabel('Level [cm]')
subplot(223),plot(t,X_est(3,:)),title('Tank 3'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
subplot(224),plot(t,X_est(4,:)),title('Tank 4'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
```

Appendix 9: MPC unconstrained

MATLAB code for MPC unconstrained with integral action

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with MPC Control with Integral action (Using unconstrained
Algorithm)

clear all;
clc;close all;

% Defining the common parameter to Minimum and non-minimum phase of
% quadruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm2
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % volts/cm
g=981;% cm/sec2

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nMPC (Unconstrained)Control of Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
Option=input('\nYour Choice:');

if Option==1 % minimum phase

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.8;
h4=1.40;
x0=[h1;h2;h3;h4];

% initial pump voltages in Volts
u10=3;
u20=3;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
```

```
% from how the valves are set prior to an experiment. Condition is
% 1<y1+y2<2.  (y1+y2=1.3)
y1=0.70;
y2=0.60;

elseif Option==2 % Non-minimum phase

% initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;
x0=[h1;h2;h3;h4];

% Initial pump voltages in Volts
u10=3.15;
u20=3.15;
u0=[u10;u20];

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. Condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;
TimeConstantOfFourTank=[T1 T2 T3 T4];

% Model Development, This is the linear Model based on
% xdot = Ac x +Bc u
% y=Dc x
% the below model is Continues Time model
A=[-1/T1,0,A3/(A1*T3),0;
    0,-1/T2,0,A4/(A2*T4);
    0,0,-1/T3,0;
    0,0,0,-1/T4];

B=[y1*k1/A1,0;
    0,y2*k2/A2;
    0,(1-y2)*k2/A3;
    (1-y1)*k1/A4,0];

D=[kc,0,0,0;
    0,kc,0,0];

%Discretizing the model, Convert from continuous- to discrete-time model
ts=0.1;%sampling time in sec, either 0.1 or 1 used
[A,B,D]=c2dm(A,B,D,zeros(2),ts,'zoh'); %  discretization
```

```
% Reading the size of the Discrete model, these size are used to create
% augmented model
n=size(A,1); % Reading the number of Rows, State variable
r=size(B,2); % Reading the number of Columns, Inputs
m=size(D,1); % Reading the number of Rows, Outputs



% Augmented Model Development or Extended state space model in terms of
% deviation variable
At=[A zeros(n,m); D eye(m,m)];
Bt=[B;zeros(m,r)];
Dt=[D eye(m,m)];

L=8; %Prediction horizon

% Weighting matrices
Q=100*eye(m); % Output Error weight matrix
              % As per the size of output, 'm' is the size or number of
              % inputs

R=0.1*eye(r); % rate of change in control action weight matrix
              % As per the size of input, 'r' is the size or thr number of
              % inputs

% Calculation parameters need for MPC like HdL, OL and OLB (Toeplitz matrix
% and Observability matrix) from a predefined m file called ss2h
[HdL,OL,OLB]=ss2h(At,Bt,Dt,zeros(m,r),L,0); % Note sending the extended
                                            % state space model
FL=[OLB HdL];

% Make extended weight matrix based on length of Prediction Horizon
% This m file is made by David Di Ruscio
Qt=q2qt(Q,L);
Rt=q2qt(R,L);

% Variables need for Unconstrained MPC
H=FL'*Qt*FL+Rt;

% Simulation Horizon
N=250;

% Generating a random set point (Reference Signal rk)
% from a predefined m file made by David Di Ruscio
rand('seed',0);
ref=[14*ones(N,1)+0.1*prbs1(N,70,70) 12*ones(N,1)+0.1*prbs1(N,70,70)];
% based on this reference signal, the initial height is also taken near to
% it.

% Calculation of Kalman Gain
G=eye(4);
Q1=10*eye(4);
R1=0.01*eye(2);
[Ke, Pp,Pe,E]=dlqe(A,G,D,Q1,R1); % given the covariance of the noise,
                                 % kalman gain Ke can found

% Initial States
r1=[14.2;12.3]; % initial height of tank 1 and tank 2 in cm
Hd=D*inv(eye(4)-A)*B;
us=inv(Hd)*r1;
```

```
xs=inv(eye(4)-A)*B*us;% Initial height of all four tank
x=xs;xold=x;
u=us;uold=u;
x_est=xold; % For kalman filter initial estimated value to be old value
yold=D*x;


%Control Loop
for k=1:N-L
    y=D*x_est; % Height of Tank 1 and tank 2 based on measured value of x
               % this value is obtained to use in kalman filter estimation
               % error and out value of height in tank 1 and tank 2.

    % Kalman filter Algorithm
    xp=xold; % set initial (apiroi) predicted state estimate
    yp=D*xp; % measurement update model
    ep=y-yp; % estimator error
    xp=xold+Ke*ep; % Corrected state estimate
    x_est=[xp(1);xp(2);xp(3);xp(4)];


    % Make the extended reference vector, r_(k,L)
    rf =ref(k+1:k+L,:); % reference signal based on prediction horizon
    ref_predhori=rf(1,:)';
    for i=2:L
        ref_predhori=[ref_predhori;rf(i,:)'];
    end

    % Compute MPC control
    xt=[x_est-xold;yold];
    pl=OL*At*xt;
    f=FL'*Qt*(pl-ref_predhori);


    %Unconstrained MPC
     duf=-inv(H)*f;
     u=u+duf(1:r); % new control signal
     uold=u; % transfering the current ctrl to old ctrl signal for next
loop

    % Storing variables for plotting
    Y(k,:)=y'; U(k,:)=u';DeltaU(k,:)=duf(1:r)'; X_est(:,k)=x_est; XX(:,k)=x;

    % Storing the current values as old values for next loop
    xold=x_est;
    yold=y;

    % Non linear model simulation with estimated states or predicted state
    % estimate update , last step of kalman filter (step 4)
    f(1)=(-a1*sqrt(2*g*x_est(1)) +a3*sqrt(2*g*x_est(3)) +y1*k1*u(1))/A1;
    f(2)=(-a2*sqrt(2*g*x_est(2)) +a4*sqrt(2*g*x_est(4)) +y2*k2*u(2))/A2;
    f(3)=(-a3*sqrt(2*g*x_est(3)) + (1-y2)*k2*u(2))/A3;
    f(4)=(-a4*sqrt(2*g*x_est(4)) + (1-y1)*k1*u(1))/A4;
    f=[f(1);f(2);f(3);f(4)];
    x_est=x_est+ts*f;

    % updation based on x measure value
    f1=((-a1*sqrt(2*g*x(1)))/A1)+(a3*(sqrt(2*g*x(3)))/A1)+((y1*k1*u(1))/A1);
    f2=((-a2*sqrt(2*g*x(2)))/A2)+(a4*(sqrt(2*g*x(4)))/A2)+((y2*k2*u(2))/A2);
```

```
    f3=((-a3*sqrt(2*g*x(3)))/A3)+(((1-y2)*k2*u(2))/A3);
    f4=((-a4*sqrt(2*g*x(4)))/A4)+(((1-y1)*k1*u(1))/A4);
    f=[f1;f2;f3;f4];
    x=x+ts*f;
end


t=1:N-L;  % Time stamp for plotting

figure(1) % Plotting the result for tank 1
plot(t,[Y(:,1) ref(1:N-L,1)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time [s]')
title('MPC UnConstrained Simulation: Output level y_k and reference r_k for
tank 1')
legend('y_k','r_k')

figure(2) % Plotting output for tank 2
plot(t,[Y(:,2) ref(1:N-L,2)]);grid on
ylabel('y_k and r_k [cm]');xlabel('Discrete time[s]');
legend('y_k','r_k');
title('MPC UnConstrained Simulation: Output level y_k and reference r_k for
tank 2')

figure(3) % plotting Control Signals of U for both tanks
subplot(211), plot(t,U(1:242,1)), grid on, ylabel('u1_k [V]')
axis([0 250 -5 8.2])
title('Tank 1: control input, u_k')
subplot(212), plot(t,U(1:242,2)), grid on, ylabel('u2_k [V]')
xlabel('Discrete time [s]')
title('Tank 2: control input, u_k')
axis([0 250 -5 8.2])


figure(4) % plotting Control Signals for delta U for both tanks
subplot(211), plot(t,DeltaU(:,1)), grid on, ylabel('\Delta u1_k [V]')
axis([0 250 -5 1.4])
title('Tank 1: \Delta U control signal ')
subplot(212), plot(t,DeltaU(:,2)), grid on, ylabel('\Delta u2_k [V]')
xlabel('Discrete time [s]')
title('Tank 2: \Delta U control signal ')
axis([0 250 -5 1.4])


figure(5) % Plot of Estimated and Measured levels
subplot(221),plot(t,X_est(1,:),'b'),grid on, hold on
plot(t,XX(1,:),'r'),title('Tank 1'), xlabel('Time[s]'),ylabel('Level [cm]')
legend('Level Est.','Level Meas.')
subplot(222),plot(t,X_est(2,:),'b',t,XX(2,:),'r'),title('Tank 2')
grid on
legend('Level Est.','Level Meas.'), xlabel('Time[s]'),ylabel('Level [cm]')
subplot(223),plot(t,X_est(3,:)),title('Tank 3'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
subplot(224),plot(t,X_est(4,:)),title('Tank 4'), legend('Level Est.')
grid on
xlabel('Time [s]'),ylabel('Level [cm]')
```

Appendix 10: Writing and Reading from Excel file for minimum phase, MPC

MATLAB coding for MPC (Constrained and Unconstrained) controller with integral action for minimum phase system, reading from excel file and plotting.

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to plot and compare the various parameter for MPC with
% minimum phase system

clc
close all

% Tank 1 , Output Level comparision
ref_tank1=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','a1:a242');
Y_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','c1:c242');
Y_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','i1:i242');
Y_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','o1:o242');
t=1:242;
figure(1)
plot(t,[ref_tank1,Y_tank1_Uncon,Y_tank1_conalgo,Y_tank1_conifelse]);
legend('Reference','Unconstrained','Constrained
(Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 1 Comparision of Level for MPC with minimum Phase')
grid on

% Tank 2  , Output Level comparision
ref_tank2=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','b1:b242');
Y_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','d1:d242');
Y_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','j1:j242');
Y_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','p1:p242');
figure(2)
plot(t,[ref_tank2,Y_tank2_Uncon,Y_tank2_conalgo,Y_tank2_conifelse]);
legend('Reference','Unconstrained','Constrained
(Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 2 Comparision of Level for MPC with minimum Phase')
grid on

% Tank 1 , Control signal comparision
U_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','e1:e242');
U_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','k1:k242');
```

```
U_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','q1:q242');
figure(3)
plot(t,[U_tank1_Uncon,U_tank1_conalgo,U_tank1_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 1 Comparision control signal U,  MPC with minimum Phase')
grid on

% Tank 2 , Control signal comparison
U_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','f1:f242');
U_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','l1:l242');
U_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','r1:r242');
figure(4)
plot(t,[U_tank2_Uncon,U_tank2_conalgo,U_tank2_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 2 Comparision control signal U,  MPC with minimum Phase')
grid on

% Tank 1 , change in Control (delta U)signal comparison
delU_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','g1:g242');
delU_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','m1:m242');
delU_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','s1:s242');
figure(5)
plot(t,[delU_tank1_Uncon,delU_tank1_conalgo,delU_tank1_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 1 Comparision of chnage in control signal (\Delta U),  MPC with
minimum Phase')
grid on

% Tank 2 , change in Control (delta U)signal comparison
delU_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','h1:h242');
delU_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','n1:n242');
delU_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlymin.xlsx','t1:t242');
figure(6)
plot(t,[delU_tank2_Uncon,delU_tank2_conalgo,delU_tank2_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 2 Comparision of chnage in control signal (\Delta U),  MPC with
minimum Phase')
grid on
```

Appendix 11: Writing and Reading from Excel file for non minimum phase, MPC

MATLAB coding for MPC (Constrained and Unconstrained) controller with integral action for non minimum phase system, reading from excel file and plotting.

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 04-March-2013
%% MATLAB file to plot and compare the various parameter for MPC with
% non non-minimum phase system

clc
close all

% Tank 1 , Output Level comparison
ref_tank1=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','a1:a242');
Y_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','c1:c242');
Y_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','i1:i242');
Y_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','o1:o242');
t=1:242;
figure(1)
plot(t,[ref_tank1,Y_tank1_Uncon,Y_tank1_conalgo,Y_tank1_conifelse]);
legend('Reference','Unconstrained','Constrained
(Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 1 Comparison of Level for MPC with non-minimum Phase')
grid on

% Tank 2  , Output Level comparison
ref_tank2=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','b1:b242');
Y_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','d1:d242');
Y_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','j1:j242');
Y_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','p1:p242');
figure(2)
plot(t,[ref_tank2,Y_tank2_Uncon,Y_tank2_conalgo,Y_tank2_conifelse]);
legend('Reference','Unconstrained','Constrained
(Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('Level [cm]')
title('Tank 2 Comparison of Level for MPC with non-minimum Phase')
grid on

% Tank 1 , Control signal comparison
U_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','e1:e242');
U_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','k1:k242');
```

```
U_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','q1:q242');
figure(3)
plot(t,[U_tank1_Uncon,U_tank1_conalgo,U_tank1_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 1 Comparision control signal U,  MPC with non-minimum Phase')
grid on

% Tank 2 , Control signal comparison
U_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','f1:f242');
U_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','l1:l242');
U_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','r1:r242');
figure(4)
plot(t,[U_tank2_Uncon,U_tank2_conalgo,U_tank2_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('U [V]')
title('Tank 2 Comparision control signal U,  MPC with non-minimum Phase')
grid on

% Tank 1 , change in Control (delta U)signal comparison
delU_tank1_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','g1:g242');
delU_tank1_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','m1:m242');
delU_tank1_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','s1:s242');
figure(5)
plot(t,[delU_tank1_Uncon,delU_tank1_conalgo,delU_tank1_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 1 Comparison of change in control signal (\Delta U),  MPC with
non-minimum Phase')
grid on

% Tank 2 , change in Control (delta U)signal comparison
delU_tank2_Uncon=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','h1:h242');
delU_tank2_conalgo=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','n1:n242');
delU_tank2_conifelse=xlsread('C:\Users\Annamalai\Google
Drive\Thesis\code\final\MPConlynonmin.xlsx','t1:t242');
figure(6)
plot(t,[delU_tank2_Uncon,delU_tank2_conalgo,delU_tank2_conifelse]);
legend('Unconstrained','Constrained (Algorthim)','Constrained(ifelse)')
xlabel('Discrete time [s]'),ylabel('\Delta u [V]')
title('Tank 2 Comparision of chnage in control signal (\Delta U),  MPC with
non-minimum Phase')
grid on
```

Appendix 12: PI Controller

MATLAB code for PI controller for quadruple tank process

```
% Course: System Control Engineering (Master of Science Degree)
% Masters Thesis 2013: Discrete LQ Optimal Control with Integral Action Vs
MPC
% College: Telemark University College,Porsgunn, Norway
% Writer: Ramanathan Annamalai (113824), Date: 05-04-2013
%% MATLAB file to simulate quadruple tank system for minimum and Non-
minimum
% phase with PI Control (Based on RGA Analysis)



clear all;
clc;close all;

% Defining the common parameter to Minimum and non-minimum phase of
% quaruple tank process based on K.H.Johansson May 2000
% Cross sectional Area of all four tanks in cm2
A1=28;
A2=32;
A3=28;
A4=32;

% Cross sectional Area of all four tank outlets in cm2
a1=0.071;
a2=0.057;
a3=0.071;
a4=0.057;

kc=1; % volts/cm
g=981;% cm/sec2

% taking input from user for minimum or non minimum phase tank datas
fprintf('\n\nPI (Constraints for U only) Control of Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
Option=input('\nYour Choice:');
fprintf('\n1 : Constraints for U  \n2 : No Constraints '); % for
constraints
Option1=input('\nYour Choice:');

if Option==1 % minimum phase chosen

%initial heights in cm for all four tank
h1=12.4;
h2=12.7;
h3=1.8;
h4=1.40;

% pump constants in cm3/Vs
k1=3.33;
k2=3.35;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. condition is
% 1<y1+y2<2.  (y1+y2=1.3)
```

```
y1=0.70;
y2=0.60;

%minimum phase PI parameters
kp=[8 12]; Ti=[10 4];

u1_init=3; u2_init=3;

elseif Option==2 % Non-minimum phase

% initial heights in cm for all four tank
h1=12.6;
h2=13;
h3=4.8;
h4=4.9;

% pump constants in cm3/Vs
k1=3.14;
k2=3.29;

% flow constants, The parameter y1 and y2 is between 0 and 1 are determined
% from how the valves are set prior to an experiment. condition is
% 0<y1+y2<1.  (y1+y2=0.77)
y1=0.43;
y2=0.34;



% non-minimum phase PI parameters
kp=[15 -.12]; Ti=[100 250];

u1_init=3.15; u2_init=3.15;
end

% Calculating Time constants using T=A(sqrt(2h/g))/a, this formula obtained
% from paper of (K.H.Johansson May 2000)
T1=A1*(sqrt(2*h1/g))/a1;
T2=A2*(sqrt(2*h2/g))/a2;
T3=A3*(sqrt(2*h3/g))/a3;
T4=A4*(sqrt(2*h4/g))/a4;

% Model Development, This is the linear Model based on
% xdot = Ax +Bu
% y=Dx
% the below model is Continues Time model
A=[-1/T1,0,A3/(A1*T3),0;
    0,-1/T2,0,A4/(A2*T4);
    0,0,-1/T3,0;
    0,0,0,-1/T4];

B=[y1*k1/A1,0;
   0,y2*k2/A2;
   0,(1-y2)*k2/A3;
   (1-y1)*k1/A4,0];

D=[kc,0,0,0;
   0,kc,0,0];
```

```
% Discretizing the model, Convert from continuous- to discrete-time model
ts=0.1;%sampling time in sec, either 0.1 or 1 used
[A,B,D]=c2dm(A,B,D,zeros(2),ts,'zoh'); %  discretization

N=2000; % Simulation Time

r1=[h1;h2];



% initial states
u_init=[u1_init;u2_init];
x_init=[h1;h2;h3;h4];
u_old=u_init; x_old=x_init; x=x_init;
y_old=D*x_init; e_old=r1-y_old;
t0=0;
umax=5;
umin=0;

%Simulation with PI controller
for k=1:N
    % creating the reference signal tank 1 and tank 2
    if k>=500&&k<1300; r1(1)=12.2; end
    if k>=300 && k<1100; r1(2)=12.4; end
    if k>=1100; r1(2)=12.8; end
    if k>=1500 r1(1)=12.5; end
    y=D*x;  % calculating the output
    e=r1-y; % find the error

    %for minimum phase process (u1,y1), (u2,y2) pairing based on RGA
Analysis
    if Option==1
       u1=u_old(1)+kp(1)*(e(1)-e_old(1))+kp(1)*ts*e(1)/Ti(1);
       u2=u_old(2)+kp(2)*(e(2)-e_old(2))+kp(2)*ts*e(2)/Ti(2);
       u=[u1;u2];

    %for non-minimum phase process (u1,y2) and (u2,y1) based on RGA
Analysis
    elseif   Option==2
             u2=u_old(1)+kp(1)*(e(1)-e_old(1))+kp(1)*ts*e(1)/Ti(1);
             u1=u_old(2)+kp(2)*(e(2)-e_old(2))+kp(2)*ts*e(2)/Ti(2);
             u=[u2;u1];
    end

    if   Option1==1
         %Implemting Constraints using ifelse loop for U for tank 1
         if u(1,1)>umax
         u(1,1)=umax;
         elseif u(1,1)<umin
         u(1,1)=umin;
         end

        % implemting constraints for second Input signal U for tank2
        if u(2,1)>umax
        u(2,1)=umax;
        elseif u(2,1)<umin
        u(2,1)=umin;
        end
    end
    % Storing the current values as old values for next loop
```

```
    u_old=u; y_old=y; x_old=x; e_old=e;

    % Storing variables for plotting
    Y(k,:)=y'; U(k,:)=u'; R(k,:)=r1'; X(k,:)=x';

    % update the model
    f(1)=(-a1*sqrt(2*g*x(1)) +a3*sqrt(2*g*x(3)) +y1*k1*u(1))/A1;
    f(2)=(-a2*sqrt(2*g*x(2)) +a4*sqrt(2*g*x(4)) +y2*k2*u(2))/A2;
    f(3)=(-a3*sqrt(2*g*x(3)) + (1-y2)*k2*u(2))/A3;
    f(4)=(-a4*sqrt(2*g*x(4)) + (1-y1)*k1*u(1))/A4;
    f=[f(1);f(2);f(3);f(4)];
    x=x+ts*f; t0=t0+ts;
end
t=1:N;
figure(1) % Plot of Tank1 , U and Output
subplot(211), plot(t,U(:,1)), grid, ylabel('u1_k [V]')
title('Tank 1: control input, u_k');
xlabel('Discrete time [s]'), ylabel('Input [V]')
if Option1==1
   legend('Umax: 5 and Umin: 0');
end
subplot(212),plot(t,[Y(:,1) R(:,1)])
title('Output level y_k and reference r_k for tank 1'), grid on
legend('y_k','r_k');
xlabel('Time[s]'), ylabel('Tank Level [cm]')
figure(2) % Plot of Tank level, U and Output
subplot(211), plot(t,U(:,2)), grid, ylabel('u2_k [V]')
xlabel('Discrete time [s]'), ylabel('Input [V]')
if Option1==1
   legend('Umax: 5 and Umin: 0');
end
title('Tank 2: control input, u_k')
subplot(212),plot(t,[Y(:,2) R(:,2)])
title('Output level y_k and reference r_k for tank 2'), grid on
legend('y_k','r_k');
xlabel('Discrete time [s]'), ylabel('Tank Level [cm]')
```

Appendix 13: RGA Analysis for PI controller

MATLAB code for RGA Analysis for PI controller for quadruple tank system

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Masters Thesis - 2013, Telemark University College, Norway
% Discrete LQ Optimal Control with Integral Action Vs MPC
% Writer: Ramanathan Annamalai (113824), Date: 04-04-2013
%************************************************************************
% MATLAB file for RGA analysis for PI Control of Four Tank system.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all; clc;
fprintf('\n\nRGA Analysis to decide input output pairs for PI Control of
Four Tank system')
fprintf('\n1 : Minimum Phase System  \n2 : Non - Minimum Phase System ');
choice=input('\nYour Choice:');
if choice==1
Hp=[2.6, 1.5; 1.4, 2.8]; % Minimum Phase
elseif choice==2
Hp=[1.5, 2.5; 2.5, 1.6]; % Non-Minimum Phase
end
Lambda=Hp.*(inv(Hp))'
```